FACULTY OF INFORMATION TECHNOLOGY AND ELECTRICAL ENGINEERING

Haoyu Chen

A Preliminary Study of Micro-Gesture: Dataset Collection and Analysis with Multi-Modal Dynamic Networks

Master's Thesis
Degree Programme in Computer Science and Engineering
May 2017

Chen H. (2017) A Preliminary Study of Micro-Gesture: Dataset Collection and Analysis with Multi-Modal Dynamic Networks. University of Oulu, Degree Programme in Computer Science and Engineering. Master's Thesis, 57p.

ABSTRACT

Micro-gestures (MG) are gestures that people performed spontaneously during communication situations. A preliminary exploration of Micro-Gesture is made in this thesis. By collecting recorded sequences of body gestures in a spontaneous state during games, a MG dataset is built through Kinect V2. A novel term 'micro-gesture' is proposed by analyzing the properties of MG dataset. Implementations of two sets of neural network architectures are achieved for micro-gestures segmentation and recognition task, which are the DBN-HMM model and the 3DCNN-HMM model for skeleton data and RGB-D data respectively. We also explore a method for extracting neutral states used in the HMM structure by detecting the activity level of the gesture sequences. The method is simple to derive and implement, and proved to be effective. The DBN-HMM and 3DCNN-HMM architectures are evaluated on MG dataset and optimized for the properties of micro-gestures. Experimental results show that we are able to achieve micro-gesture segmentation and recognition with satisfied accuracy with these two models. The work we have done about the micro-gestures in this thesis also explores a new research path for gesture recognition. Therefore, we believe that our work could be widely used as a baseline for future research on micro-gestures.

Keywords: RGB-D, 3DCNN, DBN, Recognition, Segmentation

TABLE OF CONTENTS

FOREWORD

This thesis was completed in Centre for Machine Vision and Signal Analysis at the University of Oulu, Finland. First, I would like to give my regards to my Supervisor, Dr. Guoying Zhao and my technical supervisor Xin Liu for their spending a great amount of time in assisting the thesis work. If it is not Dr. Guoying Zhao, I could not finish this thesis on time.

I would also like to thank the whole group of CMVS at the University of Oulu, I learnt a lot with their help. I will give my special thanks to Mr. Jiawei Zhang who led me through a rich Master study period and gave me plenty of help in both life and study aspects.

Second, I would also like to thank my friends for their help and accompanies during the 2 years' life in Oulu. Besides, I am grateful to the host country, Finland, for proving a good and free chance for education.

Finally, yet most importantly, I want to express my gratitude deeply to my parents for their endless love and encouragement.

Oulu, 21.05.2017

Haoyu Chen

ABBREVIATIONS

RGB	Red-Green-Blue
RGB-D	Red-Green-Blue-Depth
MG	Micro-Gesture
GMM	Gaussian Mixture Model
HMM	Hidden Markov Model
DBN	Deep Belief Network
CNN	Convolutional Neural Network
RBM	Restricted Boltzmann Machine
GRBM	Gaussian Restricted Boltzmann Machine

1. INTRODUCTION

1.1. Background

Human action and body gesture analyzing task from video sequences is a popular research field in computer vision, and there has been significant progress in static gesture recognition [1], trajectory gesture recognition [2] and continuous gesture recognition [3] during the recent years.

However, despite all the progress, there are still major challenges in gesture segmentation and recognition issues. One of those challenges is that researchers designed all those gestures and actions, so participants just performed the gestures on purpose, the gesture sequences containing continuous gestures performed by participants in spontaneous states are rarely proposed or considered.

During people's communication process, there are many cues could be observed such as facial behaviors, gestures, body movements, voice and speech characteristics, even physiological indicators (e.g., heart rate, blood pressure and skin conductance). All these cues could be classified as verbal cues (e.g., words, language) and nonverbal cues (e.g., gestures, body movements) [59]. Analyses by [60] indicated that the liars will produce significantly more nonverbal behaviors inconsistent with the context or content of their words than truth tellers do which means that nonverbal behaviors are meaningful for human affect research. Furthermore, based on the research of [12], the nonverbal behaviors, such as expressions and body gestures, could be sorted into directly delivered cues and repress cues. The repress cues are also called micros for behaviors performed spontaneously, thus we use micro-gestures to stand for gestures performed spontaneously. It is proved that micros could offer leakage of human emotions to the prediction of deception or truthfulness. Therefore, it is necessary to research micro-gestures for human behavior analysis and human emotion estimation. Furthermore, the suitable segmentation and recognition algorithms based on micro-gestures are in need.

Based on this situation, we propose a gesture dataset that contains gestures performed by participants all in spontaneous states, called MG dataset, and summarize the properties of MG dataset that differ from the gestures designed beforehand. To the fact that gesture recognitions tend to use high-level features, while Hinton and others [46] sparked a resurgence of neural networks and deep neural architectures which have been proved to be an effective approach for high-level features extracting from a set number of input samples. Then we implement two kinds of neutral networks as the DBN and 3DCNN to the dataset to test their performances on the MG dataset. We also improve these two architectures by taking the properties of MG into account to get a better result.

1.2. Related Terms Review

Here several related terms about micro-gestures and analysis process are introduced for readers to better understand the thesis.

Recorded sequences are video or skeleton joint sequences of participants. Each sequence lasts for around 15 minutes and contains the performance of the participant in this 15 minutes. The sequence includes a series of neutral states and gesture states.

Neutral states are states where the participant does not perform any movement or action in the recorded sequence. Here it should be noticed that, if the participant performs a pose while this pose lasts for a long time, it will still be a neutral state instead of a gesture.

Gesture states are states where the participant does some gestures or actions in the recorded sequence. Based on our definition of micro-gestures, gesture states will contain several kinds of micro-gesture states with one non-microgesture state. It means we will consider both micro-gestures and non-microgestures in the gesture states, and we regard all the gestures that are not micro-gestures as non-microgestures.

Micro-gestures are gestures performed by participants spontaneously, the further illustration of micro-gestures could be seen in later discussion.

Non-microgestures are gestures performed by participants on purpose. In the case of our thesis, a gesture should be either a micro-gesture or a non-microgesture. It means that micro-gestures and non-microgestures together to form the set of gesture states.

Segmentation is a procedure that obtains neutral states and gesture states from the recorded sequence in this thesis.

Recognition is a procedure that classifies the states into corresponding labels like micro-gestures or non-microgestures.

1.3. Research Objectives and Contribution

The lack of an accessible micro-gestures data has impeded the research a micro-gesture dataset, a well-established massive database of micro-gestures is in need. To solve this problem, our first target is building a micro-gesture dataset of a sufficient number of participants during an interactive game. We will also analyze state-of-the-art algorithms as the inspirations of building models for micro-gestures segmentation and recognition. Based on the analysis, our second target is to modify the existing recognition methods and implement them into MG database. Our last target is improving the performances of these algorithms by introducing optimized methods that could fit the MG dataset properties.

Our contributions are discussed as follows. First, we propose a novel MG dataset with labeled classes and organized structure. Second, modified architectures of two sets of neural networks DBN and 3DCNN are established for micro-gesture segmentation and recognition problems. Third, we carry out experiments to evaluate the performance of these two architectures. In addition, an approach for extracting neutral states in the HMM alignment is explored.

1.4. Structure of This Thesis

The structure of this thesis is as follows. Chapter 2 introduces the concept of micro-gestures, and it is the main research goal of this thesis. Some basic terms of RGB-D data and an overview of the databases with existing state-of-the-art methods are given. In Chapter 3, we review the collection process, labeling methods, and problems when building the MG dataset, and introduce the composition of the dataset. In Chapter 4, we present the DBN-HMM structure for skeleton data on this MG dataset to achieve the recognition of micro-gestures. Next, in Chapter 5 we present a detailed method using the 3DCNN model to realize the micro-gesture segmentation and recognition

with RGB video and depth data. Chapter 6 provides detailed experimental steps for micro-gesture segmentation and recognition with skeleton joints using the DBN-HMM model and the RGB-D data using the 3DCNN model. Particularly, a new method is explored to extract neutral states. Then, the experimental results are analyzed, and methods were introduced to improve the performance. Chapter 7 provides some concluding remarks of the whole work in this thesis and directions of micro-gesture for future work.

2. MICRO-GESTURE AND STATE OF THE ART

The aim of this chapter is to give a description and a definition of micro-gestures. We introduce the reader to some basic terms given about micro-gesture, although it is an immature field of image analysis. Then, a brief introduction of various gesture and action datasets settled by predecessors will be given, and we will introduce three RGB-D datasets related to our work. At last, we summarize several novel approaches used for the gesture recognition issue in recent years.

2.1. Micro-Gesture

As human action recognition research being developed mature, the more detailed analysis could be achieved, such as human gesture recognition that has become a new research topic in human action recognition in recent years. Human gesture must gain plenty of attention for its practical application domains like camera surveillance, human computer interaction, lie detector, artificial intelligence, etc. However, since micro-gesture is an emerging research field of image processing, there are few official definitions of this term. Combined with the definitions from some related fields [5, 6], such as human action and gesture [7, 8, 9], micro-expression [10, 11, 12] and psychology [13, 14], here we illustrate the definition of the micro-gesture by ourselves.

Although it is true that everyone thinks and moves differently, there is underlying unity of all the human behaviors. For instance, when people are nervous, upset, under press or telling a lie, they will behavior not as normal, and sometimes have an irresistible desire to pick at superficial body parts (as in obsessive nail-biting, scratching hair, etc.) According to this phenomenon, the definition of micro-gesture is that a spontaneous, involuntary body movement of humans according to inner emotions happened. They usually occur in high-stake situations, where people really care about something. In contrast to the normal gesture, the main property of micro-gestures is that, it is an involuntary movement that people even do not realize themselves.

In our definition, illustrative hand gestures, sign language hand gestures are not micro-gestures, as people perform these movements on purpose, or they try to use these actions to express their affects intentionally. Meanwhile, gestures like touching noses and scratching heads are micro-gestures, because people do these gestures without purposes of delivering affect information. The further details of labeling of micro-gestures will be illustrated in Section 3.3.

2.2. RGB-D Data and Kinect

In many fields and research directions like video understanding, behavior understanding, surveillance, lie detection, hidden emotion mining, and human-computer interaction, methods of recognizing human actions from video data are needed. Nowadays, the means to represent human body gestures and actions information becomes more and more variable. Depending on different types of input information, the recognition process may be designed and organized in different strategies. Among these different input types, RGB-D data is a very vital one. This thesis focuses on gesture recognition given 4D information. Here is the explanation of two vital terms, RGB-D and Kinect.

RGB-D Data and 4D Data

The initial advantage of RGB-D data over RGB data is that it could offer a precise location of the real world. It is beneficial for researchers to track body gestures of humankind in 3D. RGB-D data is short for red, green, blue and depth data. It represents an image with red, green, blue channels and a depth value of each point in the real world. Based on the concept of RGB-D, the definition of 4D data in this thesis could be given as a data type of four-dimensional information with RGB-D for the first three dimensions and temporal domain for the fourth dimension respectively.

Since video and image are all 2D data, it cannot give some 3D information directly, such as depth value, tracked pose and even the skeleton joints of human beings. Traditionally, estimating such depth related information from an image frame sequence is a difficult task, many facilities and software methods have been developed to obtain the depth related information. For the facilities and devices, the most famous device is MoCap system introduced by Muller [61] and his team that can acquire 3D joints data with hard-wired features. It has been used widely for indexing and motion tracking data and capturing 3D joints data. On the other hand, methods perceiving 3D information from 2D images also have been proposed, such as depth perception with a stereo system that utilizes the principle of disparity to obtain depth data from an image frame sequence. However, these techniques are not enough for processing 4D data in real-time.

Kinect

It is the advent of Microsoft Kinect [15, 16] that brings a novel approach to obtain the joint features of human skeleton in real time. It can also reach a satisfying accuracy. A surge has been caused by it to develop processing methods for recognizing and analyzing action and gesture from depth related data. Data like 3D images and skeletal joints positions can be researched widely. Microsoft Kinect is a motion sensing input device invented by Microsoft that can serve for Xbox and Windows PCs. It enables users to control and interact with their PC or play station without the need for a real controller. The interaction is achieved through a more natural user interface using gestures and spoken commands.

In this thesis, we used Kinect software development kit for Windows released by Microsoft. This SDK allows us to write Kinect applications in C++ with Visual Studio IDE. In addition, the SDK includes Windows-compatible PC drivers for Kinect device and three main features: raw sensor streams, skeletal tracking, and advanced audio capabilities. Here, by using features of raw sensor streams and skeletal tracking, we can access the low-level streams from both depth sensor, color camera sensor. Tracking the skeleton joints of one or two people's movement can also be achieved (to avoid interference between two Kinects, we record one participant by only one Kinect).

Figure 1. Kinect V2. This figure shows the real image of Kinect V2. It has an RGB camera and 3D depth sensor, as well as a multi-array MIC.

In our experiment, the version of the Kinect we used is Kinect V2 as it is shown in Figure 1. The parameters of it compared to Kinect V1 are listed below:

Table 1. A parameter comparison of Kinect V1 and V2

Feature	**Kinect V1**	**Kinect V2**
Color Camera	640×480@30 fps	1,920×1,080@30 fps
Depth Camera	320×240	512×424
Max Depth Distance	~4.5 m	~4.5 m
Min Depth Distance	40 cm	50 cm
Horizontal Field of View	57 degrees	70 degrees
Vertical Field of View	43 degrees	60 degrees
Tilt Motor	Yes	No
Skeleton Joints Defined	20 joints	25 joints
Full Skeletons Tracked	2	6
USB Standard	2.0	3.0
Supported OS	Win7, Win8	Win8

From the comparison in Table 1, we know that the resolution and the number of joints recorded are improved in Kinect V2. The further detailed setup of Kinect and experimental environments will be introduced in Section 3.2.

2.3. Related Work

Among numerous action and gesture related databases in machine vision field, there have been both 2D and 3D gesture or action databases for body movement and hand or body gesture research. With the Kinect and other depth record facilities being popular, 3D action databases and depth image databases have attracted more research interests as its practical function for solving reality problems. Here we will introduce the related work of gesture and action recognition from two aspects as databases and recognition methods.

2.3.1. Action and Gesture Database

After doing a wide survey of the databases related to human actions and body gestures, we achieved a general understanding of the action and gesture recognition process in computer vision field. Notice that human actions differ from body gestures or micro-gestures. To some extent, body gestures are more precise, and the recognition is finer than the action recognition. Human actions are more about the action event recognition and classification instead of fine actions and gestures. A compilations survey for

human actions and body gesture datasets can be found in Table 2. Different action and gesture classes are provided in it.

Table 2. Different databases and properties

Dataset(year)	Class	Size	Properties
KTH 2003	Action classes = 6.	600 videos (192 training, 192 validations, 216 testing). Resolution = 160x120. Black and white videos. Static camera.	Homogeneous indoor/ outdoor backgrounds. Performed by 25 persons, 4 scenes, and 6 verbs. Provided annotations: labeled temporal segments.
WEIZMANN 2005	Action classes = 10.	90 videos (evaluate by leave one out cross validation). Resolution = 180 x 144. Static camera.	Homogeneous outdoor backgrounds. Also provides irregular versions for robustness experiment.
UCF-Sports 2008	Action classes = 9.	182 videos (evaluate by leave one out cross validation). Resolution = 720 x 480. Static camera.	Actions collected from various sports in broadcast television channels.
Hollywood 2008	Action classes = 8.	430 videos (219 training, 211 testing). Resolution = 400-300x300-200 (varies between videos).	Short sequences from 32 movies. Cluttered background and occlusions between persons.
HMDB 51 2011	Action classes = 51.	6 849 videos (3 splits of 70% training and 30% testing is provided by author). Resolution = 320x240.	Videos collected from various sources such as movies, YouTube and other public databases. High diversity between videos.
MSR Action3D 2010	Action classes = 20.	4020 videos. Resolution = 640×480 of the depth map 23,797 frames of depth maps.	the 20 actions were chosen in the context of using the actions to interact with game consoles.
ChaLearn2014 2014	Gesture classes = 20.	940 video sequences. Each performed by a single person and composed of 10 to 20 gesture instances, totaling about 14,000 individual gestures.	The average length of gestures is 39 frames, the minimum frame number for a gesture is 16, and the maximum frame number is 104.

In the following sections, three well-known gesture datasets for action and gesture recognition will be introduced as well as their properties.

ChaLearn Challenge 2014

The ChaLearn database is one of the most popular and well-known human gesture corpuses, especially the ChaLearn 2014 database. The ChaLearn database is used for ChaLearn Challenges [17, 18] that is a worldwide competition series. ChaLearn 2014 organized three parallel challenge tracks on Human Pose Recovery on RGB data, action/interaction spotting on RGB data, and gesture spotting on RGB-D data. Here we mainly focus on track three that is Gesture Recognition. In this track, more than 14,000 gestures are drawn from a corpus of 20 Italian sign gesture categories. The goal of this track is to implement multi-modal machine learning of a set of 20 Italian gestures performed by different users, with the aim of recognition of users'

independent continuous gestures. An example of the visual frames of each modality is shown in Figure 2.

Figure 2 Samples of the RGB-D gesture-spotting track. Frames from four modalities are captured at the same time.

Here, we can see that the human-computer interaction gestures are research objects of this challenge. Meanwhile, training machines to recognize these 20 different kinds of gestures of human and their underlying emotions is the research target. This channel follows a previous challenge with the same theme: ChaLearn Multi-modal Gesture Recognition 2013. However, in this new edition after two years, more labels are provided with more than 900 samples. Hence, the ChaLearn Gesture database in 2014 has become one of the most important data sources to evaluate RGB-D hand and body gesture recognition algorithms.

The ChaLearn Gesture database used Kinect V1 to record data frames by Matlab, where RGB, depth and user mask modalities are all 640 × 480-pixel images. The skeleton joint positions are saved as matrixes according to the joints sequence and timing.

Microsoft Research Cambridge-12 Kinect Gesture Dataset

Another famous 3D body gesture database is the Microsoft Research Cambridge (MSR)-12 Kinect action database [19], it should be noticed that this dataset is different from MSR hand gesture 3D database [20]. Figure 3 shows the difference between the two databases, the research object of MSR-12 is the whole-body gesture while that of MSR hand gesture 3D database is only the hand region. According to the definition of micro-gestures in Section 2.1, instead of the single hand action areas, the whole-body movements and actions are focused on this thesis. MSR-12 database introduced here consists of video sequences of human movements and actions, represented as body part locations, accompanied with 12 designed gestures. It contains coordinates of 20 skeleton joints estimated using the Kinect pose estimation pipeline [21]. Those body gestures are mainly performed by two-hand movements.

Figure 3. Difference between gestures and hand gestures. For MSR-12 database, it focuses on the whole-body gesture as the left figure shows. However, MSR hand gesture database only focuses on the hand region as the right figure shows.

There are also some other action/activity data sets collected by MSR such as MSR Daily Activity data set [9], MSR Action Data Set, MSR Action Data Set 2. They all provide human actions as designed gestures in the RGB-D domain.

WEIZMANN dataset

The databases discussed above are all RGB-D databases collected by Kinect. There are also quite a lot classic gesture and action databases for human-being action and gesture recognition, such as WEIZMANN and KTH databases. Since human actions in video sequences can be regarded as shapes of a moving torso and protruding limbs undergoing articulated motion [22], human actions and movements could be regarded as three-dimensional shapes induced by the shapes in the space-time volume. A recent approach put forward by Gorelick [23] was adopted for analyzing 2D shapes and generalize it to deal with volumetric space-time-action shapes [24]. In this database description, space-time features such as local space-time saliency, action dynamics, shape structure and orientation are proved useful for action recognition, detection, and clustering.

Figure 4. WEIZMAN database. Participants performed actions and movements in front of a noisy background.

This database consists of 90 low-resolution video sequences (180×144, with frame rate at 50 fps) from nine different participants. Each participant performs 10 natural actions as it is shown in Figure 4. The actions include ‘run’, ‘walk, ‘skip’, ‘jumping-jack’, ‘jump-forward-on-two-legs’, ‘jump-in-place-on-two-legs’, ‘gallop sideways’, ‘wave-two-hands’, ‘wave one-hand’ and ‘bend’.

All these databases have an obvious property that is that all the actions or body gestures were designed and performed on purpose. It can be seen that a spontaneous gesture dataset for the micro-gesture recognition is needed. In the next chapter, we will introduce our MG database which is designed to meet this demand.

2.3.2. Action and Gesture Recognition Method

Besides the databases investigation, a survey of novel approaches of action and gesture recognition [25, 26, 27, 28, 29] in computer vision field in recent years is given. The classical methods which are always used as baselines have been investigated as well. Moreover, these methods can be sorted into two directions. One direction is to extract efficient features of human actions and body gestures, the other one is to train classification and recognition models. A summarization of classic methods of gesture and action recognition can be found in Table 3.

Table 3. Recognition method and Feature extraction methods for baseline setting

Recognition method		**Feature extraction methods**	
1 SVM	Support Vector Machines	1 LF	Local Feature
2 LCSS [62]	Longest Common Subsequence	2 HOG [64]	Histogram of Oriented Gradient
3 NNC	Nearest Neighbor Classification	3 STIP [66]	Space time interesting point
4 ISA [63]	Independent Subspace Analysis	4 SIFT [68]	Scale invariant feature transform
5 STS [22]	Space Time Shape	5 MFSK [71]	Mixed Features of Sparse Key Points
6 PT [84]	Prototype Trees	6 DTF [67]	Dense Trajectories Feature
7 PM [65]	Product Manifolds	7 SMoSIFT [34]	Sparse Motion SIFT
8 TBM [70]	Temporal Bayesian Model	8 MBH [73]	Motion Boundary Histograms
9 HMM [87]	Hidden Markov Model	9 MHI [74]	Motion History Image
10 DTW [69]	Dynamic Time Warping	10 HOJ3D [75]	Histograms of 3D Joints
11 CNN [88]	Convolution Neural Network	11 FCM [76]	Feature Covariance Matrices
12 ANN [86, 90]	Artificial Neural Network	12 KF [77]	Kinematic feature
13 DNN [89, 91]	Deep Neural Network	13 MEI [78]	Motion Energy Image
14 GMM	Gaussian Mixture Model	14 SURF [79]	Speeded-Up Robust Features
15 BoVW [72]	Bag of Visual Word		
16 MIL [80]	Multiple Instance Learning		
17 SLA [81]	Sparse Linear Approximation		
19 ERT [82]	Extremely Randomized Trees		
20 RNN [85]	Recurrent Neural Networks		
22 LDA [83]	latent dirichlet allocation		

In Table 3, several classical gesture recognition methods are listed. They are sorted into recognition methods and feature extraction methods. Besides, several novel approaches for action and gesture recognition in recent years are studied during the related work investigation. A summarization of these methods can be found from Table 4. For the feature extraction aspect, efficient local features were explored by Jun [34] with the bag of words for gesture recognition from RGB-D data. A method called one-time point annotation is also used for feature processing in gesture recognition by Long [37]. A method using the multi-scale energy-based global ternary image for 3D gesture recognition is introduced by Liu [32]. Meanwhile, for learning architecture aspect, a two-stream dictionary architecture is explored by Ke [35] to achieve action recognition, both Wu [4] and Natalia [31] use multi-modal to process gesture recognition problem. Wu combines multi-model with neural networks while Natalia proposes an adaptive multi-modal for gesture recognition. Moreover, Chenxia [36] achieved unsupervised learning of action and relations with a watch-n-patches method. The work of this thesis is inspired by the work of [4].

Table 4. New recognition methods and feature extract methods in recent years.

Method	**Description**	**Publication**	**First Author**
CSMMI [30]	Class-Specific Maximization of Mutual Information for Action and Gesture Recognition	TIP	Jun Wan
ModDrop [31]	ModDrop: Adaptive Multi-Modal Gesture Recognition	PAMI	Natalia
GTI [32]	3D Action Recognition Using Multi-Scale Energy-based Global Ternary Image	CSVT	Mengyuan Liu
DDNN [4]	Deep Dynamic Neural Networks for Multimodal Gesture Segmentation and Recognition	PAMI	Wu Di
One shot learning [33]	Discovering Motion Primitives for Unsupervised Grouping and One-Shot Learning of Human Actions	PAMI	Yang Yang
Efficient Local Features [34]	Explore Efficient Local Features from RGB-D Data for One-Shot Learning Gesture Recognition	PAMI	Jun Wan
Two-Stream Dictionary [35]	Two-Stream Dictionary Learning Architecture for Action Recognition	CSVT	Ke Xu
Watch-n-Patch [36]	Watch-n-Patch: Unsupervised Learning of Actions and Relations	PAMI	Chenxia Wu
Fixed Annotation [37]	Supporting One-Time Point Annotations for Gesture Recognition	PAMI	Long-Van Nguyen

Note that in Table 4, in the publication box, we use abbreviations like this: TIP: IEEE Transactions on Image Processing; PAMI: IEEE Transactions on Pattern Analysis and Machine Intelligence; CSVT: IEEE Transactions on Circuits and Systems for Video Technology.

3. MG DATASET

3.1. Motivation for Building a Micro-Gesture Dataset

Studying micro-gestures is initial for understanding emotional behaviors of human beings, particularly when participants are involved in a stake stage. For most people, it is tough to recognize micro-gestures. Because they are often ignored in common life. One reason for this phenomenon is that micro-gestures are very trivial and short; the other reason is that during a conversation, many other communication factors are affecting the attention of the observers at the same time - for instance, the speeches, expressions of the speakers, and their intonations. Consequently, an algorithm for automatically segmenting and recognizing micro-gestures would be very beneficial for the implementation of the gesture recognition in normal daily life, as machines have the advantages for subtle change detection and goal orientated.

From the databases summarized in Section 2.3, we know that those action and body gesture datasets are not built for the gestures that occur in a spontaneous way. In another word, in those datasets, all the gestures are designed ahead of time, and these are not micro-gestures eventually. Therefore, the lack of an accessible micro-gestures data has impeded the research. A well-established micro-gesture database with sufficient samples is in need. To solve this problem, we collect micro-gestures of a sufficient number of participants during an interactive game by Kinect v2.

In the following sections, we will introduce the method we used to collect the Micro-Gesture Dataset and illustrate its composition. The performance of this MG dataset has been tested by the DBN and 3DCNN measures. The testing results are shown in Chapter 6.

3.2. MG-dataset Collection Setup and Procedure

Many methods have been implemented to build human action and body gesture databases, some of those data collection methods are described in the previous chapter. However, none of those procedures is designed for building a spontaneous micro-gesture dataset, or say, all the participants are asked to perform some certain gestures instead of communicating spontaneously in a situated environment. For this purpose, the main requirement of this data set is that, gestures should be involuntary and be collected in natural environments.

Thus, our experimental design for getting micro-gestures is settled to be a game: participants will play a game during the experiment, and they are asked to tell a lie and try not to be exposed.

3.2.1. Setup

Here we will introduce our preparation work of setting up the facilities and environments of micro-gesture acquisition experiments.

Experimental Environment

Our micro-gesture dataset was recorded in two indoor environments places. These two environments have different scenes.

For both these two scenes, the distance of the participant and Kinect is set between 1.4-2.0 meters. The participant should stand at the view range of Kinect to make sure that all the movements and points can be recorded. While a participant is talking, a Kinect fixed in front of him/her will record his/her body gesture and actions. The setup is illustrated in Figures 5 and 6 about the recording experiment.

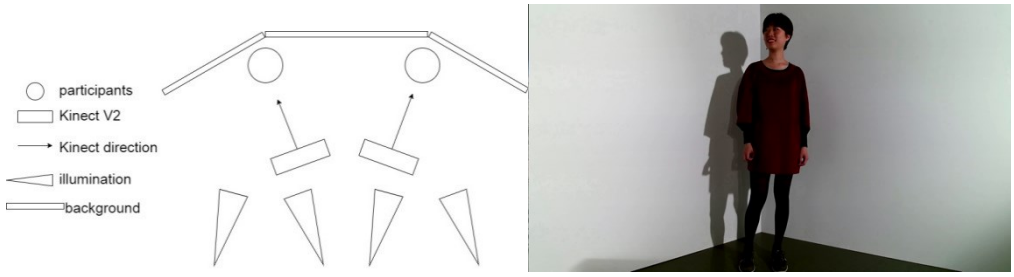

Figure 5 The experiment environment setting and a record example in CAVE lab. It has a pure white background.

The first scene is placed at CAVE lab. It is at CAVES experiment environment of ITEE faculty of the University of Oulu. It has a pure white board as a background behind the participants. Indoor illumination in this place is controlled with four incandescent lamps emitting to the participants. Although slightly stroboflash can be observed from the video, it will not affect gesture analysis. The average distance of the participant and their record Kinect is set around 3.0 meters.

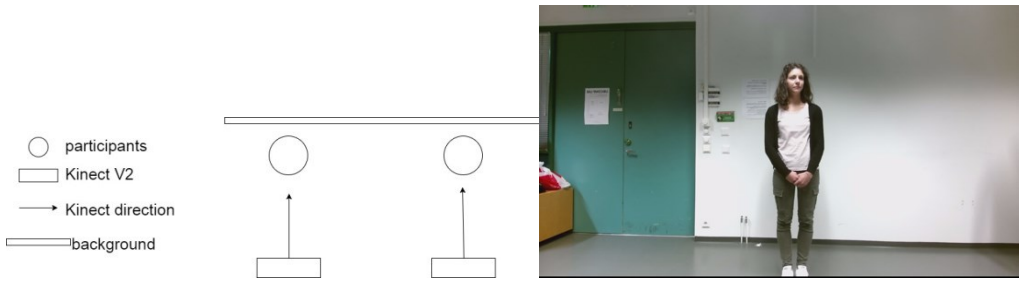

Figure 6. The experiment environment setting and a record example in UBI lab. It has a noisy background.

The second scene is placed at a common office room of UBI lab where the background is a white wall with some clutters in the scene. Indoor illumination is just normal pre-existing daylight lamps in that office room, the average distance of the participant and their record Kinect is set around 3.5 meters.

Device Setting

For the recording procedure of all the participants, Kinect V2 with 1,920×1,080 image resolution of 30-fps is used to record the whole duration of the game. Although the sample rate of Kinect V2 is 30-fps, we finally chose 28-fps to build our final micro-gesture dataset as the resolution of Kinect V2 is too high (1,920×1,080) for a normal PC to write the data into hard disk as video directly. The detailed recording procedure will be discussed in the following section.

In addition, we list the facilities required to establish the MG dataset and their types in Table 5.

Table 5. The facilities used for recording

PC	System	Windows 8 onwards (x64)
	Memory	4 GB
	USB	USB 3.0 controller dedicated to the Kinect for Windows v2 sensor
	Hard disk	256GB SSD
Software	SDK	Kinect for Windows SDK 2.0
		Kinect for Windows Runtime 2.0
	Library	OpenCV 2.4.11
	IDE	Visual Studio 2012 onwards
Kinect	Model	Model 1656
	Version	Kinect V2

3.2.2. Data Collection

Record Methods

Using Kinect APIs offered by Microsoft, a recording program for Kinect is implemented in C++, which used the Kinect 2.0 API. When it runs, it will record the video information of participants during the game. Before the experiments, participants should sign permission forms to authorize us rights to use their video information. The recorded data of the participants will be stored automatically on our server. All the data is stored in a safe place with a password needed access. Kinect Sensor has a group of sensor members, properties of the Kinect Sensor are listed in Table 6. We marked which sensor members we have used.

Table 6. Kinect Sensor properties list

Name	Description	Implemented
AudioSource	Gets the source for audio frames.	No
BodyFrameSource	Gets the source for body frames.	Yes
BodyIndexFrameSource	Gets the source for body index frames.	Yes
ColorFrameSource	Gets the source for color frames.	Yes
CoordinateMapper	Gets the coordinate mapper.	Yes
DepthFrameSource	Gets the source for depth frames.	Yes
InfraredFrameSource	Gets the source for infrared frames.	No
IsAvailable	Gets whether the Kinect sensor is available and able to retrieve frames.	Yes
IsOpen	Gets whether the Kinect sensor is open.	Yes
KinectCapabilities	Gets the capabilities of the Kinect sensor.	No
LongExposureInfraredFrameSource	Gets the source for long exposure infrared frames.	No
Sensors	Gets the list of available sensors.	Yes
UniqueKinectId	Gets the unique ID for the Kinect sensor.	Yes

With the above setting, we developed a program to collect the data. The data recording process is separated into two steps: the first step is to use Kinect to record RGB, depth, mask and skeleton information as simple binary files; the second step is to transform these binary files into real video files at 28 frame rate by Visual Studio. The recording code for this work can be found at:

https://github.com/mikecheninoulu/Kinect-v2-collecting-RGB-D-data.git

Recording Methods Comparison

According to the experimental setting, it is better to read, transform and write data at the same time. However, there are several technique issues to be considered. Since the writing speed of the experimental PC is not fast for the amount of input data during the Kinect recording process, which will cause the video frame losing problem. Then we tried several different solutions for Kinect recording. All these recording methods are designed to improve data writing speed.

The original record method is to record video without compression and data processing. It is a very original way to record and write data, and it will cause a serious frame losing issue. Averagely it can only write around four frames per second. Then we explored methods for data compression, and we used XVID compress method. XVID is an open-source compression portmanteau of coder-decoder based on MPEG-4 ISO format. We used XVID to compress the data while reading and writing to speed up disk writing speed, and this is the common method of compression in Kinect V1 recording. In this way, the number of frames written per second is averagely up to 22, but it still not enough. Finally, we decided to record RGB video into array values, and transform those array values later, and then we got 28 frames per second writing speed on average. We also used multi-threads to process the data. Its writing speed can reach 29 fps, but it requires GPU for each PC. Eventually, we chose the array record methods. Table 7 shows the sampling performances of each method.

Table 7. Kinect Sensor properties list

Solution	**Max frame rate (second)**	**Min frame rate (second)**	**Average frame rate (second)**	**Disk writing speed**	**Losing frame rate**
Video without compression	14	1	3.8	20M/s	87.3%
XVID compression with lowest distortion	25	15	20.1	4M/s	33.3%
XVID compression with lowest size	24	19	21	200k/s	30%
XVID compression with optimized setting	23	20	22	1M/s	26.7%
Only save as array file while reading	30	26	28.2	0.9M/s	6%
Use multi-thread writing	30	28	29.1	-	3%

Above all, losing frame problem exists in every record method, but the method we eventually used will bring the lowest frame-losing rate at 28 fps. To offset this loss, we set the video frame rate at 28 fps during writing array data into real video format.

Coordinate Mapper problem

When we try to map skeleton or depth coordinate to the pixels in the RGB video, there is a coordinate mapping issue. Body tracking is achieved by using the depth sensor, so the three-dimension coordinates of skeleton joint points can be aligned correctly into depth frame only. If the same skeleton joint coordinates are projected over RGB frames, the skeleton and RGB information will be incorrectly mapped as it is shown in Figure 7.

Figure 7. Mapping problem: RGB data is of two dimensions while Skeleton joints are of three dimensions.

The dimensions of the visual elements, like RGB and mask frame, are two. The representation of the visual elements is in pixels of the scene. So, converting the real-world 3D skeleton values into 2D screen pixels is needed. We use the SDK with a handy utility named CoordinateMapper for this issue. SDK CoordinateMapper is to identify a point from the 3D space and correspond it to a pixel in 2D space. Considering this issue, we record skeleton joint coordinates in both 3D format and 2D format for mapping RGB frame. The record format is introduced in Section 4.1. So, skeleton joint coordinates can be projected over color (1,920×1,080) and depth (512×424) frames without mismatching.

Collection Procedure and Game Description

The collection was held to record two participants at every turn. They were asked to play a game, and we will record their behaviors during the whole game. Before the recording experiment started, explanations of the experiment were given to participants such as how the experiment is processed and what they should do. After participants had understood how to execute the tasks, an experiment agreement forms should be signed if they allow us to use their information for research. Then, they started formal experiments in the specific laboratory and Kinects started to record the whole process.

During the whole recording process, we want to get the micro-gestures and body languages of participants, so we put the collection process into a situated scene. Then a game is designed for participants to play. The game used in the experiment is called ‘telling stories’. There are two participants for each recording experiment. For convenience, we called them X and Y. At first, they are asked to describe a story one by one. They will know the titles of all the stories beforehand. Then, participant X gets a true story. Meanwhile, participant Y gets an empty paper with several hints of the corresponding story which are shown in Table 8. Y needs to make that story to pretend that he/she gets the true one. What they should do is to prove that he/she is the one got the true story. An examiner from our research group will also ask them some details of the stories and try to distinguish them. To make it naturally, all the stories we chose are some bizarre but real things in real world, such as the largest pizza, the largest dog, etc. At one recording experiment, three stories will be discussed in CAVES lab. In UBI lab, it will be five stories. In the last story, the papers of both participants are empty. In addition, story assignment is designed. So, each participant will tell one story, and make up two fake stories in CAVES lab. While each participant will tell two

stories and make up three fake stories in UBI lab. The time is limited and task is tough for them. In this way, participants will be under high-stake states.

Explanations of the experiment were given as follows: 1) You will read several short stories, these stories are some real things in the world, such as the largest pizza, the largest dog etc. Please try to remember the stories carefully. 2) After reviewing these stories, you will find that some stories only have a title, which is empty in content. Please try to make up the story by yourself according to your imagination. 3). At last, we will discuss these stories and our examiner will ask you some questions. Please try to persuade the examiner that you have the story with content. You need to avoid being recognized you are making up a fake story.

After 15 minutes for participants to prepare and remember three or five stories, they will repeat these stories and discuss it in total around 15 minutes. There were several questions following each story asked by the examiner. Questions from the examiner always include: Could you describe the details of this thing (for instance, largest dog, longest hair)? Do you remember how large/long it is? The purpose of these questions is to make participants nervous and put them in a high-stake stage.

Table 8. Story list of the game in the experiment .

Story name	**Content of truth**	**Content of fake**	**Total word number of true story**
Pizza	The largest pizza story	How large it is; Who made it; Where and when it was made; Name of this pizza; Why this pizza was made	132
Chili	The hottest chilly story	Where it is; When it was found; How chilly it is; According to which measure standard	133
Pool	Largest swimming pool story	Name of it; How big it is; Cost; Where it is; Any detail	119
Hair	The Woman with the Longest hair story	Who she is; From where she comes; How long it is; When she started growing hair; How she takes care of her hair	152
Dog	Largest dog story	What type it is; From where it comes; How big it is; Where it is now	141

3.3. Dataset Description

The "Micro-Gesture Dataset" is designed to provide 4D data for micro-gesture studies and the development and evaluation of human micro-gestures and states recognition algorithms. Instead of asking participants to perform some specific gestures, all the gestures are collected while participants are under natural states. It contains video recordings of 52 participants communicating with others. Data is 4D information (with RGB, depth and time domain). For each participant, there are five files to record his/ her micro-gestures during the whole process. They are RGB, mask, depth, skeleton videos and skeleton information. Each recording sequence is around 15 minutes.

Participants and Data Collection

52 participants participated in the MG dataset experiment. Among those participants, 38 are male and 14 females. The participants were recruited at the University of Oulu, and the experiment was held at CAVES and UBI laboratory of ITEE faculty of the University of Oulu. All participants were either college students or research staffs. The

attribute distribution of the participants is 73% male, with an average of 25 years old (between 22-32) from 16 countries. The distribution tables are shown in Table 9. Since some participants may inhibit their behaviors, none of the participants was informed beforehand about the purpose of our research. They were asked not to leak the content of this experiment to other participants.

Table 9. Nationality distribution and gender distribution.

China	**Pakistan**	**Nepal**	**Italy**
24	6	4	3
Bangladesh	**India**	**Algeria**	**Finland**
2	2	2	1
Egypt	**Columbia**	**Ukraine**	**Portugal**
1	1	1	1
Germany	**Vietnam**	**Norway**	**Ghana**
1	1	1	1

Male	**Female**
38	14

The final MG dataset comprises of 52 sequences for all the participants (each sequence carries the whole data of a single participant) with RGB, depth, mask video sequences and skeleton data. It contains 1,451 gesture instances with 671,493 frames (approx. 6hours and 39 minutes). The average length of those samples is 15 minutes. 880 instances from 22 participants were recorded in the CAVES lab, and 571 instances from 30 participants were recorded in the UBI lab. The total numbers of the frame are 129,492 and 55,304 for CAVES and UNI respectively. The composition of MG dataset is shown in Table 10.

Table 10. MG dataset composition

Data	**Participants**	**Instances**	**Frames**
CAVES	22	880	129 492
UBI	30	571	55 304
Total	52	1 451	284 796

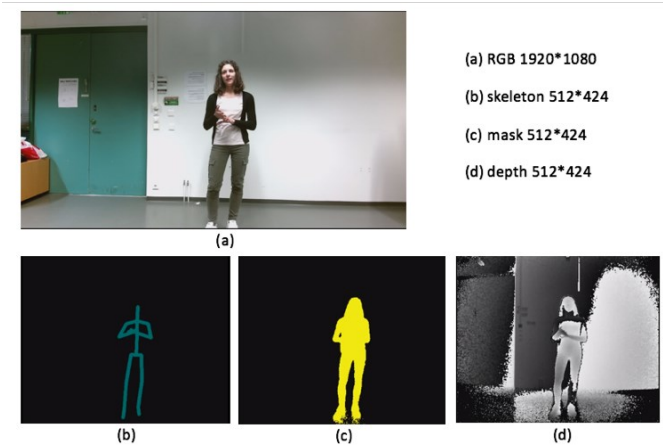

Figure 8. Sample frames of each modality at the same time. (a) RGB modality (b) skeleton modality (c) mask modality (d) depth modality, the resolution of each modality is shown in the figure.

For each participant, video recordings of their playing interactive game behaviors were collected with 4D information (RGB, depth and time domain). There are five files to record their micro-gestures during the whole process: RGB, mask, depth,

skeleton videos (at 28 frame/s, as .mp4 format) and skeleton information in text (as .csv format) as it is shown in Figure 8. Each recording is around 15 minutes.

Notice that, skeleton information contains tracks of 25 skeleton joints with coordinates, orientations and coordinate mapper points using the Kinect Pose Estimation pipeline. The body gestures are captured at a frame rate of 28Hz with around 2cm accuracy in each skeleton joint positions. The skeleton joint members are listed in Table 11 below:

Table 11. Skeleton joint values of Kinect V2

Kinect V2 members					
Member	**Value**	**Description**	**Member**	**Value**	**Description**
SpineBase	0	Base of the spine	KneeLeft	13	Left knee
SpineMid	1	Middle of the spine	AnkleLeft	14	Left ankle
Neck	2	Neck	FootLeft	15	Left foot
Head	3	Head	HipRight	16	Right hip
ShoulderLeft	4	Left shoulder	KneeRight	17	Right knee
ElbowLeft	5	Left elbow	AnkleRight	18	Right ankle
WristLeft	6	Left wrist	FootRight	19	Right foot
HandLeft	7	Left hand	SpineShoulder	20	Spine at the shoulder
ShoulderRight	8	Right shoulder	HandTipLeft	21	Tip of the left hand
ElbowRight	9	Right elbow	ThumbLeft	22	Left thumb
WristRight	10	Right wrist	HandTipRight	23	Tip of the right hand
HandRight	11	Right hand	ThumbRight	24	Right thumb
HipLeft	12	Left hip			

It should be noticed that it is different between the distributions of the Kinect V1 and V2, Kinect V2 has five more skeleton joints than that of V1 as ‘SpineShoulder’, ‘HandTipLeft’, ‘ThumbLeft’, ‘HandTipRight’, ‘ThumbRight’. In addition, ‘SpineBase’, ‘SpineMid’ and ‘Neck’ joints are introduced instead of ‘HipCenter’, ‘Spine’ and ‘ShoulderCenter’ joints. The body part allocation of corresponding joints is shown in Figure 9.

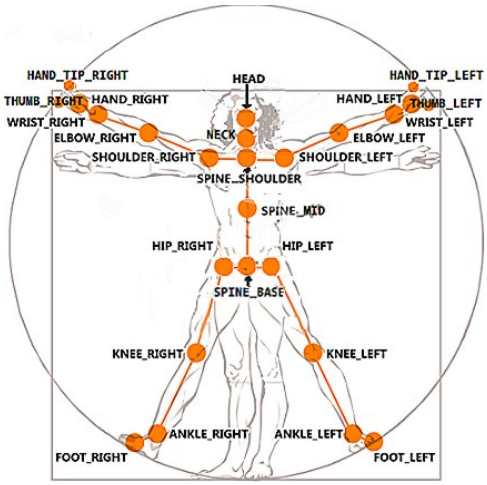

Figure 9. Skeleton Joints Distribution. The distribution of the skeleton joints is allocated as the figure shows. Each joint is mapped to a certain physical part of the body.

Also, a summary material of each experiment is gathered. It includes all the detailed data of the participants and the experiments. It contains: DATE for experiment group number and date; NUMBER for participants' number; AGE for participant age; GENDER for participant gender; NATIONALITY for participant nationality; INFO for information of participant corresponding stories; NOTE for game process and timeline to record the time when he/she is telling a lie or truth and PROBLEM for the problems during the game.

Labeling and Dataset Composition

With the definition of micro-gestures given in Section 2.1 and the properties of the data we collected, our "Micro-Gesture Dataset" has been manually labeled with two category methods. We got seven broad classes and 16 refined classes respectively as shown in Figure 10. Tabular information is included in the dataset, and it is computer-searchable.

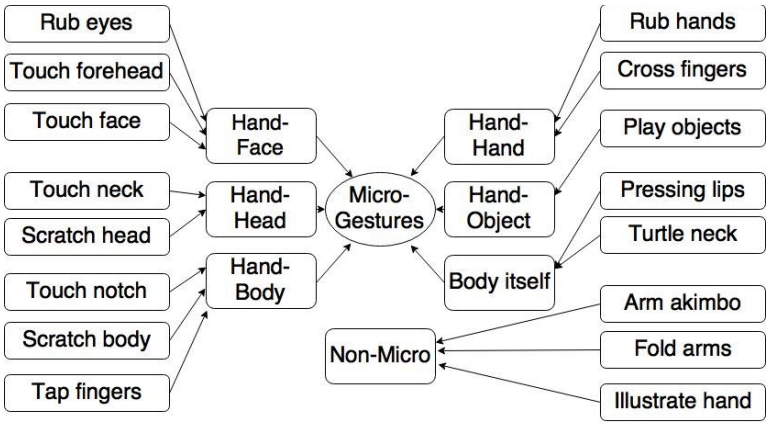

Figure 10. Micro-gesture categories and label tree of MG dataset. It contains two kinds of label strategies.

The gestures of MG dataset are categorized into two ways. One is 'fine action classes': it represents the actions of each micro-gesture in fine classes. The other one is 'coarse classes': it represents corresponding relationships between the hand and the body parts. It should be noticed that we also labeled the non-microgestures into our dataset. On the other hand, although some gestures are still in the definition of micro-gestures, the durations of them are too long to be a micro-gesture, then we filtered the gestures that are longer than 300 frames. In this way, the gesture instance number becomes 1451. A list of gesture sorts with corresponding gesture category methods along with the number of instances is given in Table 12 below:

Table 12. Micro-gesture categories and label distribution of MG dataset

ID	Fine action classes (instances)	ID	Coarse classes (instances)
1	Rubbing eyes (22)	1	Hand-Face (289)
2	Touching or scratching forehead (21)		
3	Scratching or touching other facial parts (246)		
4	Touching or scratching neck (22)	2	Hand-Head (97)
5	Scratching or adjusting head or hair (75)		
6	Rubbing hands (70)	3	Hand-Hand (118)
7	Crossing fingers (48)		
8	Playing with objects (24)	4	Hand-Objects (24)
9	Touching or scratching neck (18)	5	Hand-Body (127)
10	Scratching parts of body (83)		
11	Tapping fingers on body (26)		
12	Pressing lips (139)	6	Body itself (149)
13	Turtle neck (10)		
14	Illustrative hand gestures (557)	7	Non-Micro gesture (659)
15	Arms akimbo (25)		
16	Folding arms (77)		

From the statistical distribution of the fine action classes in Table 12, it can be seen that the distribution of these classes is not balanced. There are 557 ‘illustrative hand gestures’ while there is only 10 ‘turtle neck’. To make the categories more balanced, we sort gestures by the relationship of hand and body parts.

Dataset Contribution

The contributions of our dataset over other databases we introduced in Chapter 2 are 1). It includes spontaneous micro-gestures. Participants were placed in a game situation and they have interactive actions. They were at high stake stages, and the actions of participants are natural. 2). MG dataset includes diversity background. It can test whether the environment could affect machines of gesture recognition. We also set UBI lab as a cluttered environment to detect the reliability of recognition algorithms. 3). High resolution of our color video is achieved up to 1,920×1,080. It is beneficial for further detailed gesture recognition. 4). At last, more skeleton joints are detected by Kinect V2. It will offer more features for training.

4. GESTURE SEGMENTATION AND RECOGNITION ON SKELETON JOINTS DATA WITH DBN-HMM

The purpose of this chapter is to present the means and procedures for gesture segmentation and recognition on the skeleton joint data. These methods are based on previous work and modified to fit MG dataset.

As our MG dataset is captured and processed in Chapter 3, here we are going to introduce gesture recognition methods to test it. There are two types of input data. One is video data like RGB, depth, skeleton and mask. The other one is skeleton joints data. In this section, we will use the DBN structure to process the skeleton data and build an HMM model to train the labeled features and predict the test. In the next chapter, we will introduce a 3DCNN structure with the HMM to process RGB and depth data. The whole structure of this dataset has already been introduced in Chapter 3. Here, we will go into the details of the skeleton information of the collected data and the methods of utilizing this information.

4.1. Skeleton Joint Data

The organization of skeleton joint data is introduced in this section. Kinect V2 can detect 25 skeleton joints. To organize the skeleton joints well, a term called 'joint orientation' is introduced. The joint orientation information is to index the relationship among the skeletons. In addition, the orientation is represented by an absolute orientation according to Kinect camera coordinates. According to the alignments of those 25 joint skeletons structure, a hierarchy of bones using the skeleton joints defined by the skeletal tracking system is used here. The hierarchy uses the 'Hip Center' joint as the root and followed by the spine joints etc. The detailed structure is shown in Figure 11.

Bones are introduced here for defining the joint orientation concept. Bones are specified by the parent and child joints along the local bone. For instance, the 'Spine' bone is enclosed by the 'Hip Center' joint as its parent joints and 'Shoulder Center' joint as its child joint. With the joint orientation information, more features could be generated from skeleton joints.

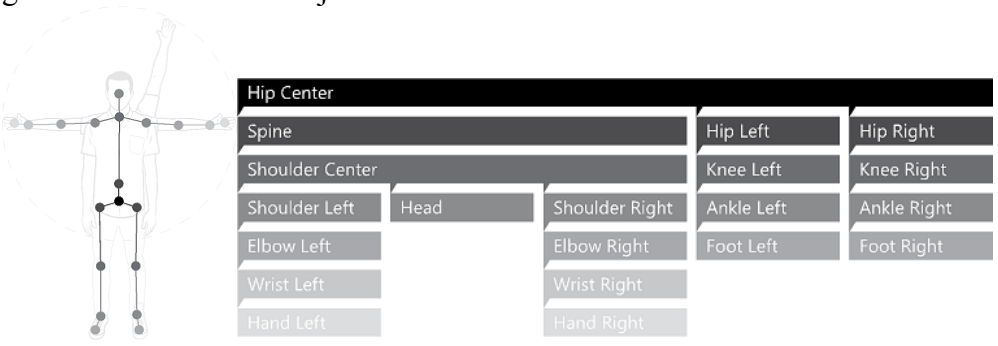

Figure 11. Bones Hierarchy. The level of the bones in the figure stands for the hierarchy position of the complete skeleton joints. The 'HipCenter' is the base of all the joints.

For each single joint, three parts of information are recorded. The first part is Joint Position covering three position values to allocate the joint positions. The second part is Joint Orientation including four values. The third part is Depth Space mapping

coordinates including two values. Table 13 below gives a brief presentation about the organization of the skeleton joints data.

Table 13. Skeleton Joint data composition

Record Number	**Number 1-3**			**Number 4-7**				**Number 8-9**	
Corresponding content	Joint Position			Joint Orientation				Depth Space Point	
Dimension	X	Y	Z	w	x	y	z	X	Y
Description of the value	Three-dimension allocation of joint position			Four-dimension allocation of joint orientation				Two-dimension allocation of joint mapping to pixel	
Note	X and Y are negative values			Some joints are self-orientated				It is from joint coordinates mapping to depth positions	

4.2. DBN Model with HMM

The primary component work of this section is giving a specific step description for applying a Deep Belief Network - Hidden Markov Model (DBN-HMM) mixture model to achieve gesture segmentation and recognition on MG dataset. Our work differs from earlier GMM-HMM and DNN-HMMs [38, 39] in two main respects. First, our research object is the micro-gesture instead of the body gesture or the action. Taking the properties of micro-gestures into account, we used several methods to improve the algorithms. Second, we analyze the influence caused by micro-gestures to the gesture segmentation and recognition which also distinguishes our work from earlier applications of DBN-HMM hybrids for gesture recognition [4, 40, 41]. The evaluation of our work is using F-score measurement concerning precision and recall, defined in [20].

The remainder of this section is organized as follows. First, we illustrate the principles of how the HMM model is used to achieve segmentation and recognition. Then, we briefly introduce the Gaussian Restrict Boltzmann Machine (GRBM) for initial weight setting. Functions of the DBN structure in the training step and the pre-training step are also introduced. After that, we describe the basic ideas how the DBN and the HMM are combined. Also, the training and decoding strategies using the DBN-HMM model is illustrated.

4.2.1. HMM Structure

Here we will introduce the process of building an HMM model fitting the micro-gestures segmentation and recognition task from our MG dataset. First, the state structure of a recorded sequence is shown in Figure 12. The HMM model will be built based on this state structure.

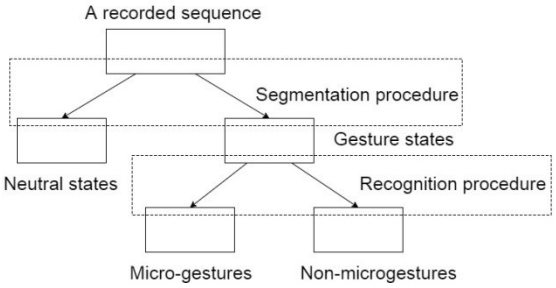

Figure 12. States structure of a recorded sequence. A recorded sequence consists of neutral states and gesture states. Gesture states contain both micro-gestures and non-microgestures.

As it shown in Figure 12, a recorded sequence consists of a series of neutral states and gesture states and these two states are arranged randomly in the sequence. Segmentation procedure is to segment neutral states and gesture states from a recorded sequence. Neutral states are states that the participant does not perform any movement or action in the recorded sequence. Here it should be noticed that, if the participant performs a pose while this pose lasts for a long time, it will still be a neutral state instead of a gesture. Gesture states are states that the participant does some gestures or actions in the recorded sequence. Thus, gesture states contain both micro-gestures and non-microgestures. Recognition procedure is to recognize different micro-gestures and non-microgestures. We assume that the neutral states of a single participant should be the same. Note that we sort all the gesture that are not micro-gestures into non-microgestures.

For the properties of micro-gestures, we propose four assumptions about the relationship between the input data (sequences of skeletal joints related information), and the corresponding classes (gesture categories). First, we assume that the discrimination of the DBN training result and the gesture classification refers to the data entity rather than the single element of the data. It means that there is underlying information of the data entity. A single element of data could not carry this kind of information. Second, we assume that, the participants will not perform gestures for the entire recording process. It means gesture states and neutral states will occur alternatively in the sequence. The third assumption is that, hidden states are generated by feature vector. So, they should be mostly unique. It means gestures should not be repetitive motions. For instance, walking includes repetitive actions, so it should not be considered. Lastly, we assume that all the micro-gestures should span in a short duration.

Based on the work of the DNN-HMM hybrid model effectively utilized on acoustic recognition [42, 43], the approach of gesture segmentation and recognition is implemented based on a Hidden Markov Model (HMM) along the temporal information [41, 44, 45]. We will first introduce our temporal HMM model for gestures, and then introduce how it is formed and the principle of it. The combination method of the HMM and the DBN will be introduced in the architecture of DBN-HMM section. Here we simply give a frame work of the HMM model. How a gesture is segmented and recognized by the HMM will be explained here.

HMM Model into Gesture Sequences

A continuous temporal observation HMM is adopted for analyzing our skeleton sequences. Basically, the average length of one participant's skeleton sequence is 15

minutes. According to the assumptions given above, we build our model for the input sequences as follows. For each sequence, it can be regarded as a sequence filled with gesture states and neutral states. So, for a single specific gesture, it follows a neutral state NS. Also, it is followed by a neutral state. Then we model the HMM based on a specific gesture and its corner states in Figure 13. Since the varieties of neutral states and gesture states are different. As varieties of gesture states are much bigger than that of neutral state, we separate a gesture state into several states n. For one specific gesture, at each time step t, one observed variable X_t for $\mathrm{t} = 1 \ldots \mathrm{T}$ will be transferred from observation layer into hidden layer. T is the total time step number. We also set a state value composition $H = \bigcup_{n \epsilon N} H_t$ including all the possible values. The values are the hidden state variable H_t from different gestures. H_t is a specific hidden state value. n is the current hidden state number. N is the total number of all the hidden states. Taking seven classes and a neutral state into account, thus, the total number of all the hidden states N should be: $N = gesture\ state\ number \times class\ number + neutral\ state\ number = 7 \times 10 + 1 = 71.$

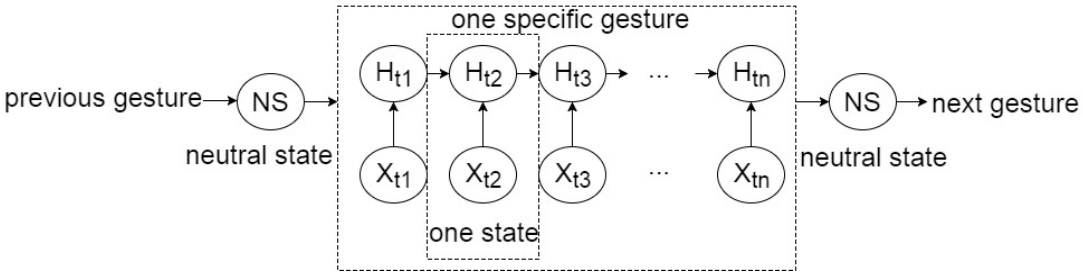

Figure 13. The HMM model for gesture sequence. One specific gesture contains several states.

A sequence of a gesture could be separated into several pose states. However, the duration of each pose state varies. What the HMM model will do is recording this variance within a Markov chain to generate a forward direction transition matrix.

Then, we are going to give a formula derivation for this HMM structure. Initially, the input training samples here are sequences of skeleton joint coordinates. We denote input sequences as x_t from visible layer state X_t. Their corresponding labels are denoted as h_t from hidden layer states H_t for $t = 1 \ldots \mathrm{T}$. The task is to learn a $decode$ function $h = decode(x)$ mapping inputs x to the corresponding label. The label is from the corpus of h. As the HMM is a generative model, principally here a joint probability distribution function $p\ (x, h)$ is needed with x and h from the training examples. Often, we have:

$$p(x, h) = p(h)p(x \mid h). \quad (1)$$

Based on this fundamental formula from the HMM model above, the full probability model of the HMM is now specified for training step as

$$p(x_1, x_2 \ldots x_T, h_1\ h_2 \ldots h_T) = p(h_1)p(x_1 \mid h_1) \prod_{t=2}^{T} p(x_t \mid h_t)p(h_t \mid h_{t-1}), \quad (2)$$

where $p(h_1)$ is the prior of the start hidden layer state. It stands for the probability of gesture start points. $p(x_t \mid h_t)$ is the observation probability or say emission distribution matrix, and $p(x_1 \mid h_1)$ is the first layer observation probability of gesture state with given hidden layer. At last, $p(h_t \mid h_{t-1})$ is the transition distribution matrix.

It stands for the possibility of one gesture state into another state. We model the emission distribution matrix of the HMM by pre-training a deep belief network. Training structure will be introduced in the next section. In addition, it is a multi-layer feed forward neural network according to the work from [47, 48]. Note that, the observation probability $p(x_t \mid h_t)$ is developed as:

$$p(x_t|h_t) = p(h_t|x_t)p(x_t)/p(h_t), \quad (3)$$

where $p(h_t|x_t)$ is the gesture state posterior probability estimated by the training result of the DBN model. The principle of it is to map a set of configurations of skeleton joint data from a certain gesture to a gesture label through neural network training. $p(h_t)$ is the prior probability of each gesture state from the composition $h = \bigcup_{n\epsilon N} h_n$. This composition includes all the possible values that finite hidden state variable H_t may take. The prior probability is also estimated from the input skeleton joint training set. For (x_t), as it does not vary with the gesture sequence and thus can be ignored. Processing the result by dividing by the prior probability $p(h_t)$ is regarded as a method called likelihood estimation scaling [49, 50, 51]. It is proved to be very effective when there is a serious bias problem in the training set. Here for the practical reason, non-gesture class has a domain weight in our training instances.

State-Transition Model in the HMM Framework

The motivation of using the HMM framework for gesture segmentation and recognition will be discussed in this part. The initial HMM model fitting the gesture sequence is achieved above. The formal proving for this HMM model is given in 4.2.1. According to formula (2), the main work of pre-training is to develop the transition matrix and emission matrix model for the further prediction target. The prediction step is achieved by decoding input skeleton joint sequence. By simulating a state transition diagram, we settled a detailed state flow chart for the state transition process which is shown in Figure 14. For each given sequence as shown in the figure, we process it as follows. By the prior probability, we enter the sequence with a certain state. For convince, we assume that the sequence starts with a gesture. Since every gesture is clipped into a continuously set of gesture states GS, the processing will go along all the gesture states of this gesture. When it comes to the end of the gesture, the processing will go from the last three states of the gesture into neutral state. When the probability of the current state is more like a gesture state than a neutral state, it will go into the first three states of the next gesture. This is how the segmentation achieved.

Furthermore, the number of state of one gesture is denoted as n. By separating a certain gesture into several states, we can use the transition relationship of those states to fit the HMM model and orient these sets of state data to a gesture label. The set of the transition matrix $p(h_t \mid h_{t-1})$ then could be set by counting numbers of the gesture state GS_n transform. The transform could be moving into either the same state (slower), state GS_{n+1}, or state GS_{n+2} (faster). This is how the recognition achieved.

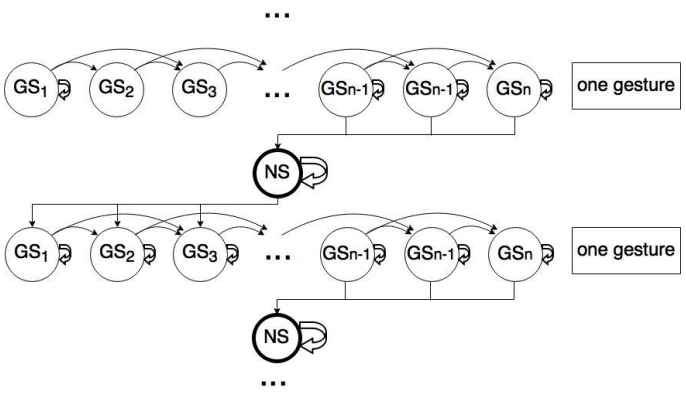

Figure 14. A transition model for gesture sequence. The last three states of a gesture will be transformed to a neutral state, and a neutral state will be transformed into the first three states of a gesture.

As it is shown above, to achieve the segmentation of gestures by this HMM model, we introduced a neutral state NS. It stands for the silence state or neutral duration between two active gesture states. From this state, we can move to the first three nodes of any gesture class, and from the last three nodes of any gesture class we can move to NS, the gesture sequences processing order could be seen in Figure 14.

4.2.2. Deep Belief Networks

According to formula (2), after transition matrix is acquired, the next work of pre-training is to develop the emission matrix model. It is for the further prediction target. The proposed DBN model here is inspired by the work of Deep Neural Networks [52], where various connectivity layers are joined to lead to high-level features from basic input data. Using the DBN architecture overcomes the limitation of just using the GRBM to model research objects. Here we will introduce both the GRBM and the DBN for network training in different steps.

Restricted Boltzmann Machines

RBMs are types of graphical model of generative stochastic artificial neural networks that can learn and produce a probability distribution from a set of inputs, and it contains a layer of hidden units and a layer of visible units. For this thesis, it is used to reflect the relationship between initial hidden states $\mathrm{h_t}$ in the first hidden layer $\mathrm{H_t}$, and number of sequences of skeleton joint coordinates $\mathrm{x_t}$ of the visible layer $\mathrm{X_t}$. We will give a formula derivation process for this RBM model in this part.

The standard type of RBM has binary-valued hidden and visible units for hidden and visible layers, and RBM consists of a matrix of weights as $\mathrm{W} = \left(\mathrm{w_{i,j}}\right)\ \mathrm{i,j} \in \mathrm{m,n}$, here m and n are the size of weight matrix. The weight matrix is associated with the connection between hidden unit h_i and visible unit v_j. In addition, that weight matrix is always accompanied by a bias weight that works as offsets a_i for the visible units and b_j for the hidden units. Given these, the energy of a configuration (the pair of Boolean vectors) (v, h) is defined as:

$$E(v,h) = -\sum_i a_i v_i - \sum_j b_j h_j - \sum_i \sum_j v_i w_{i,j} h_j, \qquad (4)$$

where $w_{i,j}$ is the matrix of visible-hidden joint weights. In the case of this thesis, the visible units v_i from the visible layer are the skeleton joint data which we have introduced above as x_t from visible layer state X_t, then we substitute v_i with x_t in formula (4), and we can get the formula for gesture case:

$$E(x,h) = -\sum_i a_i x_i - \sum_j b_j h_j - \sum_i \sum_j x_i w_{i,j} h_j. \quad (5)$$

Then, the probability of any certain pair of the visible and hidden units could be given according to the energy of that configuration (5) by:

$$p(x,h) = -\frac{e^{-E(x,h)}}{Z}, \quad (6)$$

where the normalization variable *Z* is known as the partition function:

$$Z = -\sum_{x,h} e^{-E(x,h)}. \quad (7)$$

However, the RBMs energy function of (5) is used for binary data. The gesture recognition of the skeleton joints input is typically represented as continuously feature vectors. The Gaussian-Bernoulli restricted Boltzmann machine (GRBM) is used for the state joint connection of the RBM model. After a slight modification of formula (5), the final GRBM energy function we will use in this work is then given by:

$$E(x,h) = -\frac{1}{2}\sum_i \frac{(x_i - a_i)^2}{{\sigma_i}^2} - \sum_j b_j h_j - \sum_i \sum_j \frac{x_i}{{\sigma_i}^2} w_{i,j} h_j. \quad (8)$$

Note that (8) based on the assumption that the visible units are built on a diagonal covariance Gaussian noise model. In this GRBM case, the contracture formula of GRBM is given by:

$$p(h_j \mid x) = \sigma(\sum_i w_{i,j} x_i + a_j), \quad (9)$$

where the σ stands for the logistic sigmoid and the final emission matrix of hidden layer to visible layer is given by:

$$p(x_i \mid h) = \mathcal{N}\left(x \mid \sum_j {\sigma_i}^2 w_{i,j} + b_i, {\sigma_i}^2\right), \quad (10)$$

where $\mathcal{N}(\mu, {\sigma_i}^2)$ is a Gaussian probability function with mean μ and variance ${\sigma_i}^2$. Note that in the practical training, the variance σ_i is set to be 1 in formula (10).

Deep Belief Network Pre-Training Structure

The section above described how we use formula derivation to build an RBM model. Here we will introduce how to implement deep belief network into pre-training. The reason why we introduce GRBM into training is that the initial weight for the visible and the hidden layers is needed. The DBN model is used to train and generate higher-level features for the hidden layer to represent gesture input better. The GRBM-DBN structure is shown in Figure 15. Note that GRBM model is part of the DBN training architecture, and it will be used for pre-training parameter setting.

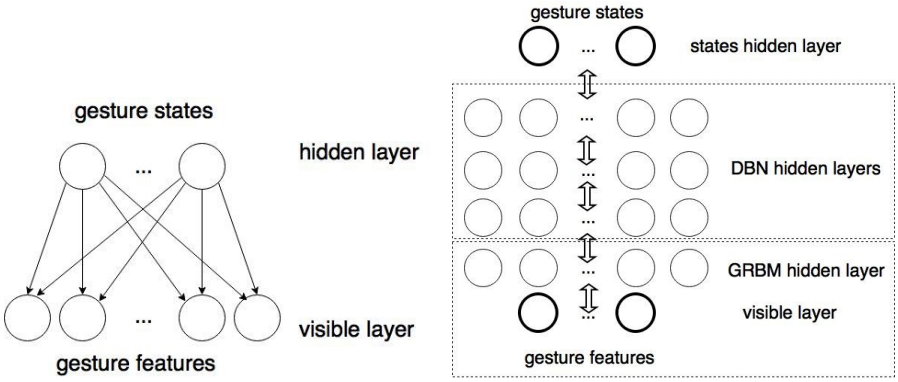

Figure 15. The left structure is GRGM model, and the right structure is the DBN model. The GRBM model is the initial layers of the DBN structure.

4.2.3. Architecture of DBN-HMM

Based on the formula (2), once we obtain the transition matrix $p(h_t \mid h_{t-1})$ for gesture states transition and emission matrix $p(x_t \mid h_t)$ from the DBN structure, the whole model training work is done. According to formula (3), to develop emission matrix $p(x_t \mid h_t)$ for decoding input sequences in further prediction part, we need to obtain both conditional probability $p(h_t \mid x_t)$ and prior for gestures states $p(h_t)$. As prior probability model $p(h_t)$ could be easily built by counting input training samples, the conditional probability $p(h_t \mid x_t)$ of the observation should be modeled. Here, Deep Belief Networks is used to train the input samples in our case. The whole structure of DBN training layers fused into the HMM model is presented below.

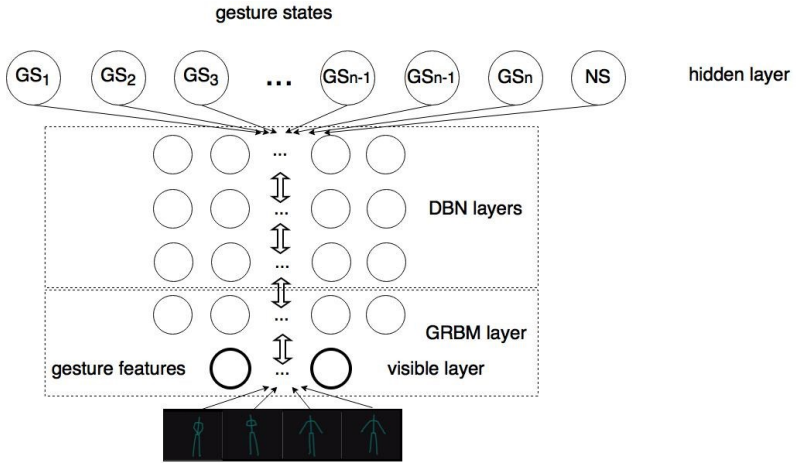

Figure 16. The DBN-HMM model for analysing gesture sequences. The input samples are skeleton joint information.

Note that the input of the observation layer is skeleton joint features. Instead of skeleton joints themselves, the skeleton images of the participants in Figure 16 is used to stand for skeleton information for better understanding. The process of extracting skeleton joint features will be introduced in the experimental part.

Finally, after we have modeled the whole DBN-HMM structure by training input samples, we can infer the gesture labeling for the input sequence according to the distribution $p(h_t \mid x_1, x_2 \ldots x_t)$ in real-time, where *t* stands for the current time for process step. Since the structure of the Hidden Markov Model is a temporal foreword

tree, decoding problem can be solved by using Viterbi algorithm. Viterbi algorithm is a max-sum algorithm that will search the most probable path of gesture states among all the possible paths efficiently [53]. The decoding result of the Viterbi algorithm is a path along the skeleton sequence of gesture states. Based on this path and the gesture state alignment of it, we can infer the class of the gesture and give the final prediction of the whole input sequence.

To build a mathematical model for Viterbi decode in our DNN-HMMs, we give the formula derivation here. First, the decoded sequence $\hat{g}$ is determined as

$$\begin{aligned} \hat{g} &= \underset{g}{\operatorname{argmax}}\, p(g \mid x_1, x_2 \dots x_T) \\ &= \underset{g}{\operatorname{argmax}}\, p(x_1, x_2 \dots x_T \mid g)p(g)/p(x_1, x_2 \dots x_T), \end{aligned} \tag{11}$$

where $p(g)$ is the prior probability of each gesture instead of gesture states, here we are going to derivate $p(x_1, x_2 \dots x_T \mid g)$ for Viterbi decode to obtain the max value for this emission probability, then we get

$$\begin{aligned} p(x_1, x_2 \dots x_T \mid g) &= \textstyle\sum_h p(x_1, x_2 \dots x_T, h \mid g)p(h \mid g) \\ &\cong max\pi(h_0) \textstyle\prod_{t=1}^{T} p(h_t \mid h_{t-1}) \prod_{t=o}^{T} p(x_t \mid h_t). \end{aligned} \tag{12}$$

With (12), we can break down the problem of solving best probability of main gesture class into solving gesture states probability with given the skeleton sequence $x_1, x_2 \dots x_T$. After all the gesture states' alignment is determined, we can infer the gestures of this sequence. In the formula (12), the observation probability $p(x_t \mid h_t)$ is obtained as

$$p(x_t \mid h_t) = p(h_t \mid x_t)p(x_t)/p(h_t), \tag{13}$$

where $p(h_t \mid x_t)$ is the gesture state posterior probability obtained by training samples with the DBN. The detailed training step is illustrated in 4.2.2.

So far, the architecture of DBN-HMM has illustrated as well as the formulation derivation. The implementation of the DBN-HMM on our MG dataset and the experimental results with analysis will be discussed in Chapter 6, as well as a performance comparison to the 3DCNN methods (in Chapter 5) on RGB and depth data.

5. GESTURE SEGMENTATION AND RECOGNITION ON RGB AND DEPTH DATA WITH MULTI-MODAL 3DCNN

The purpose of this chapter is to present the procedure of gesture segmentation and recognition with RGB and depth data on our MG dataset. Method overviews and details for the 3DCNN model are shown in the corresponding section below.

Here we will introduce gesture segmentation and recognition methods with RGB and depth data. The performance of this architecture is tested by the MG dataset. In this section, we will use a 3DCNN model to process features from RGB and depth data through multi-modal. A similar HMM structure of the DBN-HMM we introduced in Chapter 4 is implemented here to train the labeled gesture videos and predict the test samples. Moreover, we will introduce the basic principles of the 3DCNN model. How the architecture of the 3DCNN processes the RGB and depth data will be introduced.

5.1. RGB and Depth Data

In the MG dataset collection description of Chapter 3, we have given a detailed description of the data regulation of the samples. For each participant, video recordings of their playing interactive game behaviors are collected with 4D information (RGB, depth and time domain). There are five files to store their data during the whole process: RGB, mask, depth, skeleton videos and skeleton information.

We have already used the DBN-HMM model to process the skeleton information. Another important type of data is videos as RGB, mask, depth, skeleton videos. These video modalities are captured at a frame rate of 28Hz with two types of resolutions. For RGB videos, the resolution is 1 920 × 1 080, and for mask, depth, skeleton videos, the resolution is all 512 × 424. Note that skeleton videos are the graphical representation of skeleton joints coordinates which has already been processed in Chapter 4. Meanwhile, the mask videos are generated from the combination of RGB and depth data. Then the RGB videos and depth videos becomes the data carrying the most useful information. They are needed to mine further. On the other hand, by using RGB and depth videos, it is enough to model the 3D real-world distributions. With these reasons, we work on RGB and depth videos with a 3DCNN architecture to extract high-level features.

5.2. From 2D to 3D Convolutional Neural Networks

Traditionally, the convolutional neural networks (CNN) are used in 2D images doing convolution at the convolution layers to extract features from local pixels mapping the previous layer. This kind of processing focuses on the objects with two dimensions. Formally, from the work of [54], we can formula the value of a specific unit or pixel at position (x, y) at the j th feature map in the i th layer by giving

$$v_{ij}^{xy} = \tanh(b_{ij} + \sum_m \sum_{p=0}^{P_I-1} \sum_{q=0}^{Q_i-1} w_{ijm}^{pq} v_{(i-1)m}^{(x+p)(y+q)}), \quad (14)$$

where b_{ij} is the bias for the offset of the feature map, m is the index of the feature maps from $(i-1)^{th}$ P_i layer and i th layer and w_{ij}^{pq} is a weight matrix at the position (p, q) of the convolution kernel of that feature map. The height and width of the convolution

kernel for the i th layer are denoted as P_i and Q_i, respectively. In the pooling layers, the resolution of the feature maps is reduced to increase invariance to distortions on the inputs. The pooling methods could be various, and the method we used here is max-pooling. The parameters of the 2D convolutional neural networks, such as the bias b_{ij} for the unit value and the weight w_{ij}^{pq} of the kernels for the feature maps, could be trained by either supervised or unsupervised strategy [55, 56].

As demonstrated above, 2DCNN is applied on the 2D objects, and it can only compute the spatial dimensions of the features. When it comes to the video processing problems, the temporal domain information is needed for mining motion information and structures. Then the information structure of contiguous frames should be processed to extract temporal features. To this end, a 3D convolution method in the convolution stages of CNNs is proposed by [54]. With this kind of structure, both spatial and temporal dimension features could be computed. It is achieved by introducing a 3D kernel to the feature cuboid which is formed by stacking multiple feature maps from contiguous frames in the same position. By constructing this feature cuboid, the motion information is captured from multiple contiguous frames. Formally, from the work of [55], we can formula the value of a specific unit or pixel at position (x, y, t) at the j th feature map in the i th layer by giving

$$v_{ij}^{xyt} = \tanh(b_{ij} + \sum_m \sum_{p=0}^{P_I-1} \sum_{q=0}^{Q_i-1} \sum_{r=0}^{R_i-1} w_{ijm}^{pqr} v_{(i-1)m}^{(x+p)(y+q)(t+r)}), \quad (15)$$

where R_i is the length of the 3D kernel along the temporal dimension and w_{ijm}^{pqr} is a weight matrix at the position (p, q, r) of the convolution kernel of that feature map. We can get that the length of the 3D kernel along the temporal dimension R_i should be equal to the number of frames processed by a kernel each time. Figure 17 is the diagrammatic sketch of the layers mapping of 2D and 3D convolutions where the differences of 2D and 3D convolutions are shown.

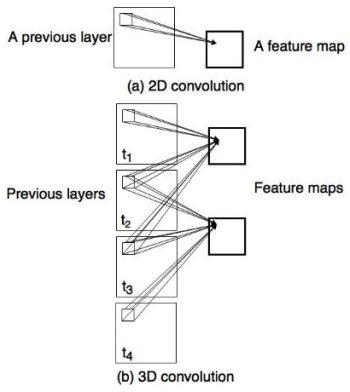

Figure 17. Different layers mapping methods of 2D and 3D convolutions. Here $\mathbf{t_1}$, $\mathbf{t_2}$, $\mathbf{t_3}$ and $\mathbf{t_4}$ stand for the current frame number, and they are continuous frames.

However, for the frame cuboids, only one type of features could be extracted from this 3D convolutional kernel. Nevertheless, the general reason of using CNNs is that the number of feature maps should be increased in late layers to generate more types of high-level features from the same pairs of lower-level feature maps. Similar to the

case of 2D convolution, this problem can be solved by applying multiple 3D convolutions with different kernels in the previous layer.

5.3. A 3D Convolutional Neural Networks Architecture

Based on the 3D convolution structure described in Section 5.2, the architectures of the 3DCNN-HMM can be derived by applying multi-layers of 3D convolution structures on the HMM model. In this section, we describe a 3D architecture of convolutional neural networks for gesture recognition on our MG dataset. This architecture consists of a set of layers including convolution, pooling, and fully connected layers. The 3D convolution is achieved by convolving the frame cuboid with a 3D kernel, the composition of the cuboid is introduced in Section 5.2. As it is shown in formula (15), traditionally, the tanh unit is always used as activation function in neural networks. However, the activation function we used here is Rectified Linear Units which is known as ReLU [56]. The reason using ReLU is that it doesn't have gradient vanishing problem as with sigmoid and tanh function. Also, it has been shown that deep networks can be trained efficiently using ReLU even without pre-training which can speed up training. Formally, we can formulate the value of a specific unit or pixel at position (x, y, t) at the j th feature map in the i th layer by giving

$$v_{ij}^{xyt} = max(0,\ (b_{ij} + \sum_m \sum_{p=0}^{P_I-1} \sum_{q=0}^{Q_i-1} \sum_{r=0}^{R_i-1} w_{ijm}^{pqr} v_{(i-1)m}^{(x+p)(y+q)(t+r)}). \quad (16)$$

The whole 3DCNN training architecture is shown in Figure 18. By implementing this 3DCNN multi-modal structure, we can fuse the RGB and depth data and get a set of high-level features. Initially, after a preprocessing phase, four types of video samples were obtained; the detailed procedure of preprocessing will be demonstrated in Chapter 6. These four sets of video samples are upper body frames in gray-scale, hand frames in gray-scale, upper body frames in depth and hand frames in depth. The resolution of all these samples is 64×64. Then we input these four types of gesture videos into the 3DCNN architecture, each of them is a four-frame sequence stacked with the size of $64 \times 64 \times 4$ for width, height, and temporal length. After applying multiple spatial convolutional kernels to the input samples with convolving, feature maps are obtained. The number and size of the kernels are 32 and 5×5. Then, the feature maps of gray-scale frames and depth frames of the upper body are fused into a single feature map by summing up. Those of the hand frames are fused in the same way. The first layer H_1 consisting of 32 feature maps is generated. It is processed by local contrast normalization (LCN) [56]. Therefore, the first layer H1 has two set of 32 feature maps for each of the hand and the upper body parts respectively. After a 3D max pooling with down sampling cubes (2, 2, 2), the second layer H_2 is generated consisting two sets of 32 feature maps with a size of $30 \times 30 \times 2$. The third layer H_3 with two sets of feature maps in $26 \times 26 \times 2$ is obtained through 64 convolution kernels with a size of 5×5. It has two sets of 64 feature maps. After processing layer H_3 with LCN and 3D max pooling with down sampling cubes (2, 2, 2) to the fourth layer H_4 with two sets of 64 feature maps in $13 \times 13 \times 1$. Then the third convolution is implemented on H_4 with 64 multiple convolution kernels resulting the layer H_5 with two sets of 64 feature maps with a size of $10 \times 10 \times 1$. The layer H_6 is then collected by 3D max pooling with down sampling cuboids (2, 2, 1), and carries two sets of 64 feature maps for hand and upper body as convolutional layer outputs

which are flattened into the layer H_7. Then, the layer H_7 has 3,200 nodes, and these nodes are fully connected to the layer H_8 with , 024 nodes. Finally, the number N of the hidden states h_n of the output layer should be $N = \text{state number} \times \text{class number} + \text{neutral state number}$.

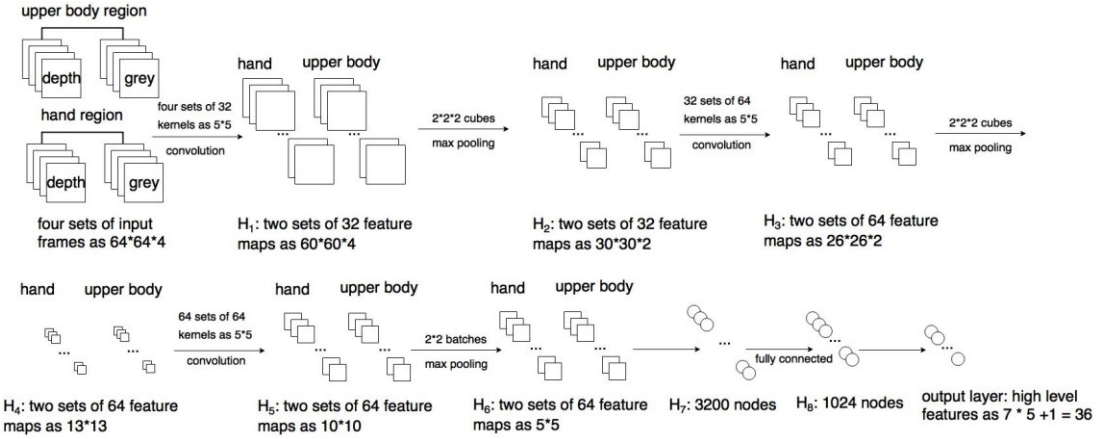

Figure 18. 3DCNN architecture used for high-level feature extraction. The input samples are four sets of RGB-D data, and the outputs are high-level features of micro-gestures.

5.4. 3DCNN Model with HMM

The primary component work of this section is giving a specific step description for applying this 3DCNN-HMM model to solve gesture segmentation and recognition problems, fitting the 3DCNN-HMM method to the properties of MG dataset. An HMM architecture for gesture segmentation and recognition has been proposed which is like the DBN-HMM structure in Chapter 4. Then we implement a 3DCNN method for the RGB and depth data feature extraction upon this HMM model by modifying the input feature from skeleton joint data into RGB-D data and feature extraction methods from the DBN model to the 3DCNN model.

The modeling procedure and formula derivation of the HMM for gesture segmentation and recognition have been illustrated in Chapter 4. Here we will emphatically introduce the basic ideas, and how the 3DCNN and the HMM are combined. The training and decoding strategies using the 3DCNN-HMM model is also illustrated.

The same as the DBN-HMM architecture, for the hidden layer of gesture states, total number N of all the hidden variable h_n hidden states should be collected by the 3DCNN training, according to the HMM hidden layer gesture states alignment, N should be $\text{state number} \times \text{class number} + \text{neutral state number}$.

Based on the formula (2), once we obtain the transition matrix $p(h_t \mid h_{t-1})$ for gesture states transition and emission matrix $p(x_t \mid h_t)$ from 3DCNN training to map gesture states to RGB-D features, the whole 3DCNN-HMM model training work is done. The transition matrix could be built in the same way as that of the DBN-HMM model. The same as the DBN-HMM architecture, to develop emission matrix $p(x_t \mid h_t)$, we need to obtain both conditional probability $p(h_t \mid x_t)$ and prior for gestures states $p(h_t)$ by the formula (3). The main step of the 3DCNN-HMM that different from the DBN-HMM is this modeling conditional

probability $p(h_t \mid x_t)$ stage. The 3DCNN is used to generate high-level features from the samples in our case. The whole structure of 3DCNN fused into a HMM model is presented in Figure 19 below.

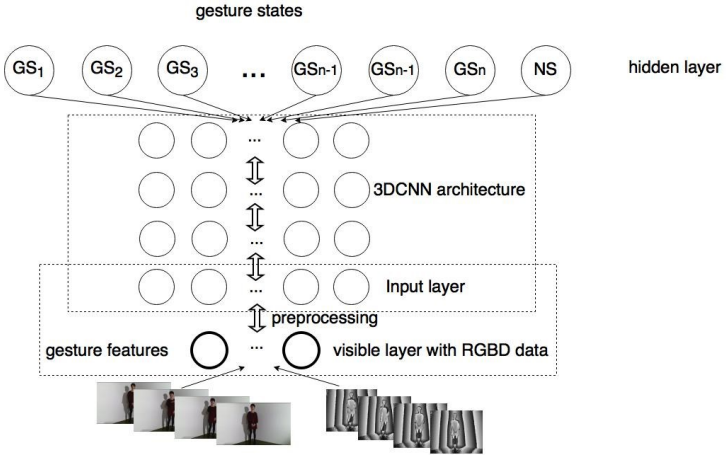

Figure 19. The 3DCNN-HMM multimodal for RGB and depth data. The RGB data and depth data will be fused in the 3DCNN architecture.

Note that the input of the observation layer will be reprocessed for the training performance of the 3DCNN; the detailed preprocessing step is introduced in experiment section. We have modeled the whole 3DCNN-HMM structure by training the RGB-D data. Then, we can infer the gesture labeling by the input sequence according to distribution $p(h_t \mid x_1, x_2 \ldots x_t)$ in real-time, where *t* stands for the current time for processing step. The Viterbi algorithm could still be used for decoding sequences here with the conditional probability trained by RGB-D data. So far, the architecture of the 3DCNN-HMM is introduced based on the DBN-HMM we introduced in Chapter 4. The detailed implementation of the 3DCNN-HMM on our MG dataset and the experimental results and the analysis will be discussed in Chapter 6. A performance comparison between two methods will be made as well.

6. EXPERIMENTAL RESULTS AND DISCUSSION

In this chapter, the implementation process is illustrated. The experimental results will be analyzed and discussed also. The experimental steps of the DBN-HMM and the 3DCNN on skeleton joint data and RGB-D data will be illustrated respectively. After that, a comparison of the performance of these two methods on our MG dataset is given. At last, an explanation of the comparison result is given. According to the micro-gesture properties, several methods are explored to improve the algorithm. Detailed experimental steps are illustrated below.

6.1. Experiments

In this section, we will illustrate the detailed experimental steps for gesture segmentation and recognition using the DBN-HMM model and the 3DCNN-HMM model separately. The feature extraction methods for skeleton joints and RGB-D data are introduced. The parameter setup for the DBN and 3DCNN for training are listed. Our micro-gesture dataset is used as the training and testing sets. The project code for these two architectures can be found at:

https://github.com/mikecheninoulu/ThesisprojectforDBN3DCNN

Features

For the skeleton joint features processing, the methods we used to are introduced as follows. With the property of 3D skeleton joints and the work of [4], we extracted the skeleton joint features in the following way. First, the 3D coordinates of N joints of a specific frame could be denoted as: $X = \{x_1, x_2 \dots x_N\}$. Then, Eigen Joints are introduced here for processing skeleton joint features. Eigen Joints are features that utilizing 3D position difference characters of joints to generate more spatiotemporal information. Eigen Joints include the posture feature f_{cc} , the motion feature f_{cp}, and offset feature f_{ci} for every single frame. The utilities and principles of those features are explained below.

The posture feature f_{cc} is used to extract static posture information of a gesture in a specific frame. It is obtained by computing differences of all the skeleton joints in that frame:

$$f_{cc} = \{x_i - x_j \mid i, j = 1,2, \dots, N; i < j\}. \quad (17)$$

The motion feature f_{cp} is used to extract static motion information from a specific frame c by computing the differences between the skeleton joints coordinates of that frame and the preceding frame p:

$$f_{cp} = \{x_i^c - x_j^p \mid x_i^c \in X_c;\ x_j^p \in X_p\}. \quad (18)$$

Moreover, the last offset feature f_{ci} is used to represent the offset feature from a specific frame c by computing the differences of skeleton joints relationship between frame c and to the initial frame I, it also can represent the overall dynamics of the gestures from the frame:

$$f_{cp} = \{x_i^c - x_j^I \mid x_i^c \in X_c;\ x_j^I \in X_I\}. \quad (19)$$

With a direction concatenation, we can get a skeleton joint feature set f_c which will be used as input data of our experiment: $f_c = \{f_{cc}, f_{ci}, f_{cp}\}$. In this way, the dimension N_f of skeleton joint input features becomes

$$N_f = \left(N_{joints} \times \frac{(N_{joints}-1)}{2} + N_{joints}^2 + N_{joints}^2\right) \times 3. \quad (19)$$

Note that before the feature extraction step, all the 3D joint coordinates are normalized by transforming them to a person-centric coordinate system. This system places the 'HipCenter' at the origin instead of the world coordinate system.

For the phase of feature extraction from RGB and depth video, the methods we used are introduced as follows. The Kinect V2 recorded RGB and depth data are of 1,920 × 1,080, and 512 × 424 resolution. However, it is not necessary to use the whole image that is very time-consuming for feature extraction. Then we crop the demand regions of the images, for instance, hand regions and upper body regions, to reduce computation complexity. Then our first step is to extract the single hand and the upper body regions from the original frames. The locations of these two regions are based on the label information of the skeleton joint distribution. The reason why we chose these two regions for feature extraction is that the most active regions of those gestures belong to hand and upper body. Note that the single hand we chose to extract is always the higher hand. It is based on the fact that two hands of a participant tend to be mirrored when two hands are used in a gesture. Then only one hand is needed for feature extraction. At the same time, if only one hand is used in a gesture, this hand must be at a higher position than the other one. Therefore, we just extract one hand region that has a higher position.

After transforming RGB videos into grey-scale frames, we preprocessed these four video samples which are upper body and hand in grey-scale and depth into frame batches with a resolution of 64 × 64. A set of normalization strategies is used for depth frames by removing the background noises with mask data, applying a median filter and subtracting mean values.

Parameters setup

We set our DBN-GRBM network model architecture as [N_X N_2 2,000 1,000 N_{GS}] which contains four layers for the whole structure. In this architecture, N_X is the dimension of the skeleton joint input observation domain, N_2 is the number of hidden units in GRBM structure, the inner layers between N_2 and N_{GS} are nodes for each layer in the DBN and N_{GS} is the number of the target output gesture states. Note that in the experimental stage, we also perform several DBN structures with different complexities to make a comparison. By using the DBN-GRBM, we can extract high-level skeleton features. In our experiments for all the datasets, the number of states is chosen as 10 for the gesture states model of the corresponding gesture class. The feed forward networks of the DBN are pre-trained with a small batch size of 100 training cases, and 100 epochs were run for pre-training. 10% of the training samples is used as validation, and another 10% of the training samples is used for testing during the fine-tuning stage.

During the training procedure of the 3DCNN architecture, dropout strategy [57] is used as the regularization approach to overcome the overfitting problem. Mini-batches of size 64 for the training are also used.

For the both architecture, 40 samples are used for training set, and ten samples are used for testing set. Note that two samples containing too few gesture labeling and are not suitable for testing. Thus, we don't use these two samples for experiments. The sample usage distribution is listed in Table 14 below:

Table 14. The sample usage distribution for the two architectures

	DBN-HMM		**3DCNN-HMM**	
	Training	Testing	Training	Testing
CAVES lab	16	5	16	5
UBI lab	24	5	24	5

6.2. Optimizations for MG Properties

Since it is a micro-gesture dataset that we implement our methods for segmentation and recognition on, we need to consider the properties of micro-gestures and improve our methods according to it. Here we first introduce the properties of micro-gestures and then illustrate the optimized methods we used to fit the model into micro-gestures.

Micro-Gesture Properties

According to the definition given in Chapter 2, we summarized several micro-gesture properties that need to be considered during the algorithm design and experiment procedure. Here four main properties that characterize micro-gestures would be useful in algorithm design. The first one is that it is possible that gestures could continuously occur instead of sandwiching a neutral state between them. For instance, after the participant performs one micro-gesture, he/she may perform another gesture instead of being back to neutral states. The second one is that the number of non-microgestures is overwhelming to that of micro-gestures. In other words, the micro-gestures are much fewer than non-microgestures. The classification bias problem during the recognition procedure should be considered. Thirdly, the active regions of a gesture tend to be upper body or even just hand parts. Lastly, another main property of micro-gestures is that they should span in a short duration relative to the durations of non-microgestures.

According to these properties of micro-gestures, we proposed optimization methods for improving the performance of the DBN-HMM and the 3DCNN methods on micro-gestures.

Optimizations for MG Properties

Based on the micro-gesture properties, we used this method to optimize the algorithm to fit MG dataset better. First, we filtered all the gesture samples in the labeling system which is longer than 150 frames, as micro-gestures. Second, the number of non-microgestures is overwhelming to that of micro-gestures, then we introduced a method called likelihood estimation scaling by processing the result with prior probability $p(h_t)$ based on the formula (3), the benefit of this method is already introduced in Section 4.2. Lastly, like the cropping method we used to process the RGB-D data, we only selected skeleton joints which belong to the upper body region. Therefore, we

only take the upper nine body joints as input for our task in the DBN-HMM training. The nine joints used as upper body region are ‘Head’, ‘SpineMid’, ‘SpineBase’, ‘ElbowLeft’, ‘WristLeft’, ‘ShoulderLeft’, ‘HandLeft’, ‘ElbowRight’, ‘WristRight’, ‘ShoulderRight’, ‘HandRight’. Note that all the results of this improved method are compared to those of original means that are listed in the experimental results section.

6.3. An Activity Level Based Detection

Besides the methods that we introduced above for feature processing, we explored a method for neutral states auto-detection and extraction. Variance factor is utilized to learn the activity level of the input sequences. The neutral states are then selected based on their activity levels. This neutral state extraction method is simple to derive and implement. It is proved to be more effective for neutral state setting than the manual labeling way according to the experimental results. Here is a schematic diagram of this extraction method.

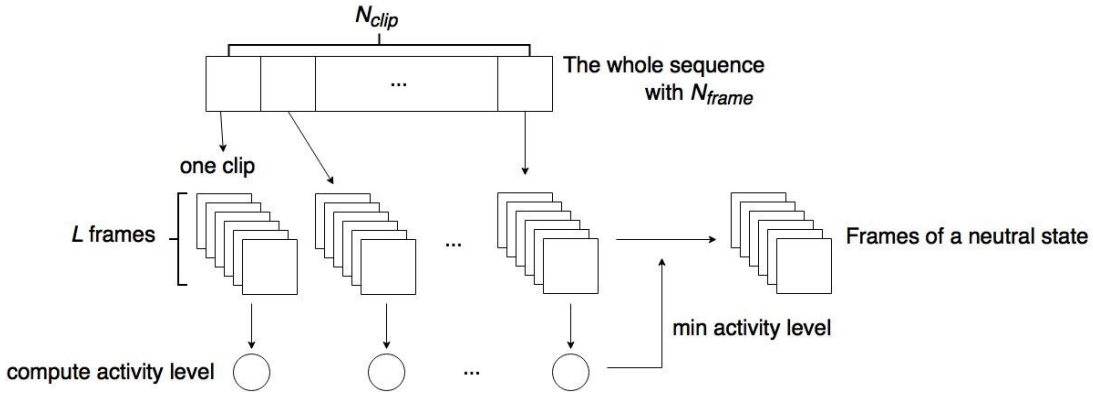

Figure 20. The neutral state extraction procedure based on the activity level. The whole input sequence will be separated into several clips, and each clip will be detected with activity level.

As it is shown in Figure 20, we denote the total frame number of the processing sequence is N_{frame}. Then we separate the sequence averagely into several frame clips. The frame length of each clip should be the same which we denote as L and the number of frame clips N_{clip} should be $N_{frame}/\ L$. After N_{clip} clips are generated, we capture the activity level of the gesture state in the current clip by summing up the variance of used joint coordinates. The activity level value for the current clip is given by

$$v\ =\ \sum_{l=1}^{L}\sum_{p}^{P}(var\left(\sum_{d=1}^{D}x_d^p\right)), \tag{20}$$

where x is the value of the joint in dimension d, and D is the whole dimension number of the joint, p stands for the current joint and P stands for the joints used. In our experiment, we chose left and right-hand joints as used joints, because hand joints are the most representative for gesture description.

In addition, this activity level based neutral state detection method can be used for labeling before training. Thus, an unsupervised learning could be achieved in the future work.

6.4. Evaluation Methods

Initially, our target is to train those two models with given samples to achieve micro-gestures segmentation and recognition. There are two sets of labeling strategies for MG dataset that are illustrated in Chapter 3: 'fine action classes' and 'coarse classes'. We will use 'coarse classes' as our classification strategy. The 'coarse classes' includes seven classes which are 'Hand-Face', 'Hand-Head', 'Hand-Hand', 'Hand-Objects', 'Hand-Body', 'Body itself', 'Non-Micro gesture'. We will use this labeling strategy to train and evaluate the samples. We used 40 sequences for training and the ten sequences for testing for both these two architectures. Note that there are two kinds of subsets of MG dataset, one is CAVES dataset, and the other one is UBI dataset, so we make training set and testing set contain samples from both sets. Then, a large number of frames will be used for training (219,073 frames) and testing (60,722 frames) which is advantageous for neural networks like the DBN-HMM and the 3DCNN-HMM. Above is the data assignment for the training and testing. The experimental results are evaluated by F-score:

$$F - score = 2\ \frac{prec(\Delta) \times rec(\Delta)}{prec(\Delta) + rec(\Delta)}, \tag{21}$$

where *prec* and *rec* stand for precision and recall and Δ stands for a specified amount of tolerated latency. Based on this formula, both the gesture segmentation and recognition results could be evaluated by *F-score.*

6.5. Experimental Results

The training and testing samples we used in the experiment are from the MG dataset. The data distribution and composition are demonstrated in Chapter 3. We will use skeleton modality to test the performance of the DBN-HMM model introduced in Chapter 4, and use RGB with depth modalities to test the performance of the 3DCNN-HMM architecture introduced in Chapter 5.

Our first experiment tests the basic performance of the DBN-HMM on MG dataset by changing sample amount assignment and set types. The DBN network model architecture is obtained by a combination of [528 2,000 2,000 1,000 71] and [528 1,000 1,000 500 71]. We chose only the CAVES set for training and testing. Such a setup achieved 0.239 F-score for recognition as shown in Table 15. Switching training set to the whole MG dataset with the same DBN-GRBM structure increased performance up to 0.275.

Table 15. Comparison of the DBN-HMM performance on difference data

Sample source		
Number of training sequences	16 of CAVES set	40 of all sets
Number of testing sequences	5 of CAVES set	10 of all sets
Highest F-score	0.288	0.474
Lowest F-score	0.209	0.214
Average F-score	0.239	0.275

Then, we compare several sets of the DBN architectures. The purpose of this step is to explore a better DBN structure for our MG dataset. As the combination of [528 2,000 2,000 1,000 71] and [528 1,000 1,000 500 71] may be too complex for the number of our MG dataset samples that may cause an over-fitting problem. Meanwhile, if it is possible that a simpler structure could be implemented on MG dataset, it will be a computationally waste to keep using the original DBN architecture. Note that in practical implements, we fuse the conditional probabilities of three models from different structures, and here we list two combinations of the DBN structures to represent.

Table 16. Comparison of the DBN-HMM performance with difference structures

Different DBN structures		
	Plan A	Plan B
Model 1	[528 2,000 2,000 1,000 71]	[528 2,000 1,000 71]
Model 2	[528 2,000 2,000 1,000 71]	[528 2,000 1,000 71]
Model 3	[528 1,000 1,000 500 71]	[528 1,000 500 71]
Highest F-score	0.474	0.595
Lowest F-score	0.214	0.186
Average F-score	0.275	0.319

As it is shown in Table 16. One structure is plan A with four-layer networks which consists of two network model with [528 2,000 2,000 1,000 71] and one network with [528 1,000 1,000 500 71]. The other structure is plan B with three-layer networks. It consists of two network model with [528 2,000 1,000 71] and one network with [528 1,000 500 71]. The following array of the model stands for the node number of each layer. For instance, in the model 1 of plan A, there are four layers, and each layer contains [2,000 2,000 1,000 71] nodes, while in the model 2 of plan B, there are three layers and each layer contains [2,000 1,000 71] nodes.

Table 17. Training time of different DBN structures

Training time of different DBN structures				
DBN structure	[528 2,000 2,000 1,000 71]	[528 1,000 1,000 500 71]	[528 2,000 1,000 71]	[528 1,000 500 71]
DBN layers	4	4	3	3
Average pre-training time (minutes)	160	110	90	50
Fine tune time (minutes)	40	30	25	18

The training time of the corresponding DBN structure is listed in Table 17. We tested the running time with Python programs, Theano and OpenCV libraries on MATE Desktop Environment 1.16.2 (GPU: NVidia Tesla K80, RAM: 12 GB). From Table 17 we can see that the three-layer DBN architecture is enough for our MG dataset. The performance of the four-layer one is slightly worse as over-fitting may have happened, the three-layer structure is more efficient for practical training as it can save huge time.

In the last experiment for DBN-HMM, we compare the performances of the two different neutral states extraction methods, one is from manual labeling, and the other one is from activity level detection method. The result is shown in Table 18.

Table 18. Comparison of different neutral state extraction methods

DBN-HMM with different neutral state extraction methods			
Number of testing sequences	manual labeled	activity level detection	activity level detection
DBN layers	4	4	3
Highest F-score	0.474	0.637	0.642
Lowest F-score	0.214	0.228	0.218
Average F-score	0.275	0.331	0.333

For the performance of the 3DCNN-HMM model on RGB-D, we use the same dataset assignment strategy for training and testing with 40 samples and ten samples respectively. Since we do not implement activity level detection methods on the 3DCNN-HMM model, the DBN-HMM results used for comparison to the performance of the 3DCNN is without activity level detection either. The comparison result is shown in Table 19.

Table 19. Comparison of the performance of the DBN-HMM and the 3DCNN

Sample source		
Number of training sequences	DBN-HMM/skeleton	3DCNN-HMM/RGB-D
Highest F-score	0.474	0.240
Lowest F-score	0.214	0.144
Average F-score	0.275	0.202

Finally, different state numbers of one gesture are tested for the DBN-HMM as Table 20 below, we can find that the performance of using ten states could be slightly better but it is not recommended as its computational complexity is much larger than that of five states.

Table 20. Comparison of the 3DCNN-HMM performance on difference gesture state number

Different gesture state number of 3DCNN-HMM		
State number	5	10
Highest F-score	0.240	0.245
Lowest F-score	0.144	0.150
Average F-score	0.202	0.209

6.6. Results Analysis and Discussion

Based on the experimental results above, the analysis is given here with the performance of the corresponding models and methods.

Here we represent five experiments for the DBN-HMM and 3DCNN-HMM models. In the first experiment, it is proved that a large quantity of training data is beneficial for deep neural network training result as the general property of using NNs is that it is capable of the considerable huge number of feature maps to generate more types of high-level features.

The second experiment proved that a simpler DBN architecture is effective to our MG dataset. From the experimental results, the reason supposed to be that the number of input sequences is not large, it can be seen that an overfitting problem may arise when we use deep layer networks (e.g. four layers or more). So, a three-layers structure is enough for training.

In the third experiment, the activity level detection method is tested. Obviously, the recognition accuracy of the DBN-HMM is enhanced. We give our explanation of that better performance: With using activity level detection, neutral states extraction could be implemented based on the state conditional character of the recorded sequence. We assume that all the neutral states of the participants are the same. However, there still exist plenty of differences in real cases, and activity level detection can be capable dealing with this diversity. For instance, for one participant, the neutral state is standing straightly, but for the other participant, the neutral state maybe folding arms. By using activity level detection method, this kind of diversity could be wiped out, because it only focuses on the gesture activity level. It means different gesture states could be absorbed as neutral states as long as participants keep the states unchanged.

In the fourth experiment, a comparison of the DBN-HMM (skeleton) with the 3DCNN-HMM (RGB-D) is given. From the result, we can learn that the performance of the DBN-HMM structure is better than the 3DCNN-HMM model. One potential reason is that the processing objects of the DBN-HMM structure are skeleton which contains more location information of gestures, the precise is more up to 2 cm. On the other hand, although a multi-modal by fusing RGB and depth data is implemented in the 3DCNN-HMM architecture, the temporal information may not be kept as max pooling is used in the procedure of the 3DCNN structure. Another reason is that many steps (such as reducing the model complexity and activity level detection) were used to optimize the DBN-HMM.

Finally, the main tricky point of micro-gestures still belongs to the fact that non-microgestures and micro-gestures are too similar sometimes. By rechecking the recognition results of the classified sequences, we can find that many non-microgestures were categorized into micro-gestures, especially the ‘hand to body’ class. How to strictly distinct non-microgestures and micro-gestures is a profound research task, as it involves not only computer vision but also psychology knowledge. Regardless, it would be the target of our future work.

7. CONCLUSION

In this thesis, we have studied gestures and micro-gestures analysis by computer vision. One key part of the work is the exploration of micro-gestures, and MG dataset collection and labeling. Furthermore, we implemented micro-gesture segmentation and recognition using two neutral network architectures.

For the study of micro-gestures, we have reviewed gesture recognition methods and gesture databases in recent years. Then a preliminary definition of micro-gestures is given. We also designed a micro-gesture dataset collection procedure and set up the MG dataset including more than 1,000 micro-gestures instances from 52 participants. Two labeling strategies are given and applied to MG dataset.

For the micro-gestures analysis, we implemented two sets of neural network architectures for segmentation and recognition task. Practical implementation of the DBN-HMM model is achieved. It can process skeleton joints data of a sample sequence. The process output is the segmentation and recognition of the micro-gestures in that sequence. Meanwhile, a multi-modal (fusing RGB and depth data) of the 3DCNN-HMM is also achieved for micro-gestures segmentation and recognition. These two methods are tested on our MG dataset.

Additionally, we explored a method as activity level detection for setting the neutral states in the HMM model. Commonly, the manual labeling neutral states method was used in designed gesture sequence. In this thesis, we try to utilize variance factor to learn the activity level of the gesture sequences. Then, the neutral states can be selected based on their activity levels. Although this neutral state extraction step is simple, it is effective in improving the experimental results.

We also presented the analysis of the experiments. Experimental result analysis of variety DBN-HMM structures is given with a different number of input samples. It is proved that few input samples with complex architectures will lead to an over-fitting problem. A comparison between the DBN-HMM and 3DCNN-HMM is given. The experimental results show that for the MG dataset, the DBN-HMM based approach is more suitable.

As future work, there are several directions. One is exploring different multi-modal architectures based on HMM, such as fusing the skeleton and RGB-D features from the DBN and 3DCNN architectures. Besides, a more regular labeling strategy should be set with a reliable micro-gesture classification. It is better to label the micro-gestures with a sufficient psychology fundamental. It would be more efficient for improving the recognition results and reasonable for future research. Another research path would be to improve the 3DCNN architecture with temporal information through the max pooling.

8. REFERENCES

[1] C. Xu & L. Cheng (2013) Efficient hand pose estimation from a single depth image. Proc. IEEE International Conference on Computer Vision: 3456-3462. DOI: 10.1109/ICCV.2013.429.

[2] Y. Wang, T. Yu, L. Shi, & Z. Li (2008) Using human body gestures as inputs for gaming via depth analysis. Proc. 2008 IEEE International Conference on Multimedia and Expo: 993-996. DOI: 10.1109/ICME.2008.4607604.

[3] J. Alon, V. Athitsos, Q. Yuan, & S. Sclaroff (2009) A unified framework for gesture recognition and spatiotemporal gesture segmentation. Proc. IEEE Transactions on Pattern Analysis and Machine Intelligence 31(9): 1685-1699. DOI: 10.1109/TPAMI.2008.203.

[4] D. Wu, L. Pigou , P. J. Kindermans, N. Le & L. Shao (2016) Deep Dynamic Neural Networks for Multimodal Gesture Segmentation and Recognition. Proc. IEEE Transactions on Pattern Analysis and Machine Intelligence 38(8): 1583-1597. DOI: 10.1109/TPAMI.2016.2537340.

[5] E. Haggard & K. Isaacs (1966) Micromomentary facial expressions as indicators of ego mechanisms in psychotherapy. Methods of research in psychotherapy. New York: Appleton-Century-Crofts: 154-165.

[6] Z. Zeng, M. Pantic, G. Roisman, & T. Huang (2009) A survey of affect recognition methods: Audio, visual, and spontaneous expressions. Proc. IEEE Transactions on Pattern Analysis and Machine Intelligence 31(1): 39-58. DOI: 10.1109/TPAMI.2008.52.

[7] Yang, M., Ahuja, N. (1998) Extraction and Classification of Visual Motion Patterns for Hand Gesture Recognition. Proc. 1998 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. DOI: 10.1109/CVPR.1998.698710.

[8] Yang, J., Xu, Y., Chen, C. (1994) Gesture Interface: Modeling and Learning, Proc. IEEE International Conference on Robotics and Automation 2: 1747-1752. DOI: 10.1109/ROBOT.1994.351340.

[9] N. Neverova, C. Wolf, G. Taylor, & F. Nebout (2014) Multi-scale deep learning for gesture detection and localization. Proc. 2014 European Conference on Computer Vision and Pattern Recognition Workshops: 474-490. DOI: 10.1007/978-3-319-16178-5_33.

[10] S. Escalera, J. Gonzlez, X. Bar, M. Reyes, O. Lops, I. Guyon, V. Athitsos, & H. J. Escalante (2013) Multi-modal gesture recognition challenge 2013: Dataset and results. Proc. ACM ChaLearn Multi-Modal Gesture Recognition Grand Challenge and Workshop: 445-452. DOI: 10.1145/2522848.2532595.

[11] J. Wang, Z. Liu, Y. Wu, & J. Yuan (2012) Mining actionlet ensemble for action recognition with depth cameras. Proc. IEEE Conference on Computer Vision and Pattern Recognition. DOI: 10.1109/CVPR.2012.6247813.

[12] P. Ekman (2009) Lie catching and microexpressions. The philosophy of deception: 118-133. DOI:10.1093/acprof:oso/9780195327939.003.0008.

[13] P. Ekman & M. O'Sullivan (1991) Who can catch a liar. American psychologist 46(9): 913-20.

[14] J. Shotton, A. Fitzgibbon, M. Cook, T. Sharp, M. Finocchio, R. Moore, A. Kipman, & A. Blake (2011) Real-time human pose recognition in parts from a single depth image. Proc. IEEE Conference on Computer Vision and Pattern Recognition: 1297-1304. DOI: 10.1109/CVPR.2011.5995316.

[15] J. Han, L. Shao, D. Xu, & J. Shotton (2013) Enhanced computer vision with microsoft kinect sensor: A review. Proc. IEEE Transactions on Cybernetics 43(5): 1318-1334. DOI: 10.1109/TCYB.2013.2265378.

[16] J. Suarez & R. R. Murphy (2012) Hand gesture recognition with depth images: A review," Proc. IEEE International Symposium on Robot and Human Interactive Communication: 411-417. DOI: 10.1109/ROMAN.2012.6343787.

[17] J. Wang, Z. Liu, Y. Wu, & J. Yuan (2012) Mining actionlet ensemble for action recognition with depth cameras. Proc. IEEE Conference on Computer Vision and Pattern Recognition: 1290-1297. DOI: 10.1109/CVPR.2011.5995316.

[18] L. Liu, L. Shao, F. Zheng, & X. Li (2014) Realistic action recognition via sparsely-constructed Gaussian processes. In IEEE Conference on Computer Vision and Pattern Recognition 47: 3819-3827, DOI: 10.1016/j.patcog. 2014.07.006.

[19] X. Yang & Y. Tian (2012) Eigenjoints-based action recognition using naive-bayes-nearest-neighbor. Proc. IEEE Conference on Computer Vision and Pattern Recognition Workshops. DOI: 10.1109/CVPRW.2012.6239232.

[20] S. Fothergill, H. M. Mentis, P. Kohli, & S. Nowozin (2012) Instructing people for training gestural interactive systems. Proc. the SIGCHI Conference on Human Factors in Computing Systems: 1737-1746. DOI: 10.1145/2207676.2208303.

[21] J. Shotton et al. (2011) Real-time human pose recognition in parts from single depth images. Proc. IEEE Conference on Computer Vision and Pattern Recognition: 1297-1304. DOI: 10.1109/CVPR.2011.5995316.

[22] S. Belongie, J. Malik, & J. Puzicha (2002) Shape Matching and Object Recognition Using Shape Contexts. IEEE Transactions on Pattern Analysis and Machine Intelligence 24(4): 509-522. DOI: 10.1109/ICCSIT.2010.5565098.

[23] M. Blank, L. Gorelick, E. Shechtman, M. Irani, & R. Basri (2005) Actions as Space-Time Shapes. Proc. IEEE Transactions on Pattern Analysis and Machine Intelligence 29(12): 2247-53. DOI: 10.1109/TPAMI.2007.70711.

[24] M.J. Black (1999) Explaining Optical Flow Events with Parameterized Spatio-Temporal Models. Proc. IEEE Conference on Computer Vision and Pattern Recognition 1: 1326-1332. DOI: 10.1109/CVPR.1999.786959.

[25] N. C. Camgo¨z, A. A. Kindiroglu,& L. Akarun (2014) Gesture recognition using template based random forest classifiers. Computer Vision-ECCV 2014 Workshops. Springer: 579-594. DOI: 10.1007/978-3-319-16178-5_41.

[26] J. Y. Chang (2015) Nonparametric gesture labeling from multi-modal data," in European Conference on Computer Vision and Pattern Recognition Workshops: 503-517. DOI: 10.1007/978-3-319-16178-5_35.

[27] G. D. Evangelidis, G. Singh, & R. Horaud (2014) Continuous gesture recognition from articulated poses. Computer Vision-ECCV 2014 Workshops. Springer: 595-607. DOI: 10.1007/978-3-319-16178-5_42.

[28] G. Chen, D. Clarke, M. Giuliani, A. Gaschler, D. Wu, D. Weikersdorfer, & A. Knoll (2014) Multi-modality gesture detection and recognition with un-supervision, randomization and discrimination. Computer Vision-ECCV 2014 Workshops. Springer: 608-622. DOI: 10.1007/978-3-319-16178-5_43.

[29] B. Liang & L. Zheng (2014) Multi-modal gesture recognition using skeletal joints and motion trail model. Computer Vision-ECCV 2014 Workshops. Springer: 623-638. DOI: 10.1007/978-3-319-16178-5_44.

[30] J. Wan & V.Athitsos (2014) CSMMI: Class-Specific Maximization of Mutual Information for Action and Gesture Recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 23(7): 3152-3165. DOI: 10.1109/TIP.2014.2328181.

[31] N. Neverova,& C.Wolf (2016) ModDrop: Adaptive Multi-Modal Gesture Recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 38(8): 1692-1705. DOI: 10.1109/TPAMI.2015.2461544.

[32] M. Liu, H. Liu (2017) 3D Action Recognition Using Multi-Scale Energy-based Global Ternary Image. Proc. IEEE Transactions on Circuits and Systems for Video Technology: 99. DOI: 10.1109/TCSVT.2017.2655521.

[33] Y. yang, I.Saleemi (2013) Discovering Motion Primitives for Unsupervised Grouping and One-Shot learning Of Human actions , gestures, and Expressions. IEEE Transactions on Pattern Analysis and Machine Intelligence 35(7): 1635-1648. DOI: 10.1109/TPAMI.2012.253.

[34] J. Wan, G. Guo (2015) Explore Efficient Local Features from RGB-D Data for One-Shot Learning Gesture Recognition. IEEE Trans. Pattern Analysis

and Machine Intelligence 38(8): 1626-1639. DOI: 10.1109/TPAMI.2015.2513479.

[35] K. Xu, X. Jiang (2017) Two-stream Dictionary Learning Architecture for Action Recognition. Proc. IEEE Proc. IEEE Transactions on Circuits and Systems for Video Technology 27(3): 567-576. DOI: 10.1109/TCSVT.2017.2665359.

[36] C. Wu, J. Zhang (2017) Watch-n-Patch: Unsupervised Learning of Actions and Relations. IEEE Transactions on Pattern Analysis and Machine Intelligence 99: 1. DOI: 10.1109/TPAMI.2017.2679054.

[37] L. Van (2016) Supporting One-Time Point Annotation for Gesture Recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence 99: 1. DOI: 10.1109/TPAMI.2016.2637350.

[38] K. P. Murphy (2012) Machine learning: a probabilistic perspective. The MIT Press.

[39] A. Acero, N. Bernstein, R. Chambers, Y. Ju, X. Li, J. Odell, P. Nguyen, O. Scholtz, & G. Zweig (2008) Live search for mobile: Web services by voice on the cellphone. Proc. 2008 IEEE International Conference on Acoustics, Speech and Signal Processing: 5256-5259. DOI: 10.1109/ICASSP.2008.4518845.

[40] D. Wu & L. Shao (2013) Silhouette analysis-based action recognition via exploiting human poses. IEEE Transactions on Circuits and Systems for Video Technology 23(2): 236-243. DOI: 10.1109/TCSVT.2012.2203731.

[41] D. Wu & L. Shao (2014) Leveraging hierarchical parametric networks for skeletal joints based action segmentation and recognition," in IEEE Conference on Computer Vision and Pattern Recognition: 724-731. DOI: 10.1109/CVPR.2014.98.

[42] S. Renals, N. Morgan, H. Bourlard, M. Cohen, & H. Franco (2014) Connectionist probability estimators in hmm speech recognition. IEEE Transactions on Speech and Audio Processing 2(1): 161-174. DOI: 10.1109/89.260359.

[43] H. Bourlard & N. Morgan (1994) Connectionist speech recognition- a hybrid approach. KLUWER ACADEMIC PUBLISHERS, Tech. Rep.

[44] D. Wu & L. Shao (2014) Deep dynamic neural networks for gesture segmentation and recognition. European Conference on Computer Vision and Pattern Recognition Workshops. 38(8): 1583-1597. DOI: 10.1109/TPAMI.2016.2537340.

[45] L. Pigou, S. Dieleman, P.-J. Kindermans, & B. Schrauwen (2014) Sign language recognition using convolutional neural networks. European Conference on Computer Vision and Pattern Recognition Workshops: 572-578. DOI: 10.1007/978-3-319-16178-5_40.

[46] G. E. Hinton, S. Osindero, & Y.-W. Teh (2006) A fast learning algorithm for deep belief nets. Neural computation 18(7): 1527-1554. DOI: 10.1162/neco.2006.18.7.1527.

[47] A. Krizhevsky, I. Sutskever, & G. E. Hinton (2012) ImageNet classification with deep convolutional neural networks. Neural Information Processing Systems: 1097-1105.

[48] A. Mohamed, G. E. Dahl, & G. Hinton (2012) Acoustic modeling using deep belief networks. IEEE Transactions on Audio, Speech, and Language Processing 20(1): 14-22. DOI: 10.1109/TASL.2011.2109382.

[49] H. Boulard & N. Morgan (1993) Continuous speech recognition by connectionist statistical methods. IEEE Transactions on Neural Networks 4(6): 893-909. DOI: 10.1109/72.286885.

[50] H. Bourlard & N. Morgan (1994) Connectionist Speech Recognition: A Hybrid Approach, ser. The Kluwer International Series in Engineering and Computer Science. Boston, MA: Kluwer: 247.

[51] S. Renals, N. Morgan, H. Boulard, M. Cohen, & H. Franco (1994) Connectionist probability estimators in HMM speech recognition. IEEE Transactions on Speech Audio Process 2(1): 161-174. DOI: 10.1109/89.260359.

[52] R. Salakhutdinov (2009) Learning deep generative models. Ph.D. dissertation, University of Toronto.

[53] C. Bishop (2006) Pattern recognition and machine learning. Springer.

[54] S. Ji & W. Xu (2013) 3D Convolutional Neural Networks for Human Action Recognition. IEEE Trans. Pattern Analysis and Machine Intelligence 35(1): 221-231. DOI: 10.1109/TPAMI.2012.59.

[55] Y. LeCun, L. Bottou, Y. Bengio, & P. Haffner (1998) Gradient-Based Learning Applied to Document Recognition. Proc. IEEE 86(11): 2278-2324. DOI: 10.1109/5.726791.

[56] A. Krizhevsky, I. Sutskever, & G. E. Hinton (2012) Using very deep autoencoders for content-based image retrieval. European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning Neural Information Processing Systems.

[57] K. Jarrett, K. Kavukcuoglu, M. Ranzato, & Y. LeCun (2009) What is the best multi-stage architecture for object recognition. Computer Vision, 2009 IEEE 12th International Conference on: 2146-2153. DOI: 10.1109/ICCV.2009.5459469.

[58] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, & R. R. Salakhutdinov (2012) Improving neural networks by preventing coadaptation of feature detectors. arXiv:1207.0580v1.

[59] Ekman P, Friesen W (1969) Nonverbal leakage and clues to deception. Psychiatry 32(1): 88-106. DOI: dx.doi.org/10.1080/00332747.1969.11023575.

[60] Ekman P, Friesen W (1974) Nonverbal Behavior and Psychopathology. The Psychology of Depression: Contemporary Theory and Research: 203-224.

[61] M. M¨uller & T. R¨oder (2006) Motion templates for automatic classification and retrieval of motion capture data. Proc. 2006 ACM SIGGRAPH/Eurographics symposium on Computer animation: 137-146.

[62] W.-T. Chan, Y. Zhang, S. P.Y. Fung, D. Ye, & H. Zhu (2005) Efficient Algorithms for Finding A Longest Common Increasing Subsequence. International Symposium on Algorithms and Computation: 665-674. DOI: 10.1007/11602613_67.

[63] Q.V. Le, W.Y. Zou, S.Y. Yeung, & A.Y. Ng (2011) Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis. IEEE Conference on Computer Vision and Pattern Recognition: 3361-3368. DOI: 10.1109/CVPR.2011.5995496.

[64] F. Baumann (2013) Action recognition with hog-of features. In 35th German Conference on Pattern Recognition: 243-248. DOI: 10.1007/978-3-642-40602-7_26.

[65] Y. M. Lui, J. Beveridge, & M. Kirby (2010) Action classification on product manifolds. In Computer Vision and Pattern Recognition. IEEE Conference on. DOI: 10.1109/CVPR.2010.5540131.

[66] I. Laptev & T. Lindeberg (2003) Motion detection and tracking using space-time interest points. Computer Systems and Applications (AICCSA), 2013 ACS International Conference on: 432-439. DOI: 10.1109/AICCSA.2013.6616419.

[67] H. Wang, A. Klaser, C. Schmid, C.L. Liu (2011) Action recognition by dense trajectories. Proc. IEEE Conference on Computer Vision and Pattern Recognition: 3169-3176. DOI: 10.1109/CVPR.2011.5995407.

[68] D. Lowe (1999) Object Recognition from Local Scale-Invariant Features. Proc. International Conference on Computer Vision. IEEE Computer Society 2: 1150-1157. DOI: 10.1109/ICCV.1999.790410.

[69] Donald J. Berndt, James Clifford (1994) Using dynamic time warping to find patterns in time series. Proc. 3rd International Conference on Knowledge Discovery and Data Mining: 359-370.

[70] M. R. Malgireddy, I. Inwogu, V. Govindaraju (2012) A temporal Bayesian model for classifying detecting and localizing activities in video sequences. Proc. IEEE Conference on Computer Vision and Pattern Recognition Workshops: 43-48. DOI: 10.1109/CVPRW.2012.6239185.

[71] P. Dollar, V. Rabaud, G. Cottrell, S. Belongie (2005) Behavior recognition via sparse spatio-temporal features. 2005 IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance: 65-72. DOI: 10.1109/VSPETS.2005.1570899.

[72] Svetlana Lazebnik, Cordelia Schmid, Jean Ponce (2006) Beyond Bags of Features: Spatial Pyramid Matching for Recognizing Natural Scene Categories. Proc. 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition: 2169-2178. DOI: 10.1109/CVPR.2006.68.

[73] H. Wang, A. Kläser, C. Schmid, C.-L. Liu (2013) Dense trajectories and motion boundary descriptors for action recognition. International Journal of Computer Vision 103(1): 60-79. DOI 10.1007/s11263-012-0594-8.

[74] G. Bradski, J. Davis (2000) Motion Segmentation and Pose Recognition with Motion History Gradients. Fifth IEEE Workshop on Applications of Computer Vision: 238-244. DOI: 10.1109/WACV.2000.895428.

[75] L. Xia, C.-C. Chen, J. Aggarwal (2012) View invariant human action recognition using histograms of 3d joints. Computer Vision and Pattern Recognition Workshops (CVPRW) 2012 IEEE Computer Society Conference on: 20-27. DOI: 10.1109/CVPRW.2012.6239233.

[76] K. Guo, P. Ishwar, J. Konrad (2009) Action recognition from video by covariance matching of silhouette tunnels. Proc. Brazilian Symposium on Computer Graphics and Image Processing: 299-306. DOI: 10.1109/SIBGRAPI.2009.29.

[77] K.M. Cheung, S. Baker, T. Kanade (2003) Shape-from-Silhouette of Articulated Objects and Its Use for Human Body Kinematics Estimation and Motion Capture. Proc. Proc. IEEE Conference on Computer Vision and Pattern Recognition: 299-306. DOI: 10.1109/SIBGRAPI.2009.29.

[78] J. Han, B. Bhanu (2006) Individual recognition using gait energy image. IEEE Transactions on Pattern Analysis and Machine Intelligence 28(2): 316-322. DOI: 10.1109/TPAMI.2006.38

[79] Bay H., Tuytelaars T., Van Gool L. (2006) SURF: Speeded Up Robust Features. Computer Vision - ECCV 2006. Lecture Notes in Computer Science, 3951. Springer, Berlin, Heidelberg.

[80] P. Viola, J. C. Platt, & C. Zhang. (2005) Multiple instance boosting for object detection. In NIPS: 1417-1426.

[81] Benson, M. W. & Frederickson, P. O. (1982) Iterative solution of large sparse linear systems arising in certain multidimensional approximation problems. Utilitas Math 22: 127-140.

[82] Geurts P, Ernst D, Wehenkel L (2006) Extremely Randomized Trees. Machine Learning 36: 3-42. DOI: 10.1007/s10994-006-6226-1

[83] David M. Blei, Andrew Y. Ng, Michael I. Jordan (2003) Latent dirichlet allocation. The Journal of Machine Learning Research 2: 993-1022.

[84] Z. Lin, Z. Jiang, L. Davis (2009) Recognizing actions by shape-motion prototype trees. Proc. Computer Vision, 2009 IEEE 12th International Conference on: 444-451, 2009. DOI: 10.1109/ICCV.2009.5459184

[85] Oriol Vinyals, Suman Ravuri, & Daniel Povey (2012) Revisiting Recurrent Neural Networks for Robust ASR. 2012 IEEE International Conference on Acoustics, Speech and Signal Processing: 4085-4088. DOI: 10.1109/ICASSP.2012.6288816

[86] W. T. Beyene (2007) Application of artificial neural networks to statistical analysis and nonlinear modeling of high-speed interconnect systems. IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems 26(1): 166-176. DOI: 10.1109/TCAD.2006.882518

[87] Y. Bengio, P. Frasconi (1995) An Input Output HMM Architecture. Advances in Neural Information Processing Systems 7: 427-434.

[88] Rongjian Li, Dong Si, Tao Zeng, Shuiwang Ji, Jing He (2016) Deep convolutional neural networks for detecting secondary structures in protein density maps from cryo-electron microscopy. Bioinformatics and Biomedicine (BIBM) 2016 IEEE International Conference on: 41-46. DOI: 10.1109/BIBM.2016.7822490

[89] H. Liao (2013) Speaker adaptation of context dependent deep neural networks. Proc. 2012 IEEE International Conference on Acoustics, Speech and Signal Processing: 7947-7951. DOI: 10.1109/ICASSP.2013.6639212

[90] O. Abdel-Hamid, H. Jiang (2013) Fast speaker adaptation of hybrid NN/HMM model for speech recognition based on discriminative learning of speaker code. Proc. 2013 IEEE International Conference on Acoustics, Speech and Signal Processing: 7942-7946. DOI: 10.1109/ICASSP.2013.6639211

[91] J. Neto, L. Almeida, M. Hochberg, C. Martins, L. Nunes, S. Renals, T. Robinson (1995) Speaker-adaptation for hybrid HMM-ANN continuous speech recognition system. Proc. Fourth European Conference on Speech Communication and Technology, Eurospeech.